Imperial College London

BEng Individual Project

Imperial College London

Department of Computing

# An Architecture to Navigate the Topical World of Instagram

*Author:*
Andreas Asprou

*Supervisor:*
Alex Carver

*Second Marker:*
Giuliano Casale

June, 2018

# Abstract

Instagram allows over 800 million users to communicate with each other using a visual vocabulary of images and videos. This visual communication includes high-quality content creators sharing content over a diverse set of niche topics specific to Instagram. Despite this fact, there are no reliable methods to navigate Instagram topically, e.g. search for content creators by niche topic. This work aims to provide a first attempt at establishing topical structure for Instagram. To achieve this, we present an architecture for classifying content creators on Instagram into their niche topics of content over a large custom-curated taxonomy. The nature of Instagram results in unique challenges to the task of topic inference and classification. For instance, the noisy, sparse and unreliable nature of text on Instagram degrades the performance of state-of-the-art topic inference, modelling and discovery approaches.

The topics of content creators inferred by this architecture are exposed to end-users through a variety of products, resulting in the additional challenge of achieving both high precision and recall classification. The proposed architecture solves the vast array of challenges by designing three independent components. The first component aims to solve the challenge of discovering the niche topics that are posted about on Instagram. To this end we design a novel topical local community detection (T-LCD) algorithm to discover niche and tightly knit communities of topically coherent content creators. These communities are utilised to enumerate the vast diversity of topics that are posted about on Instagram. Finally, using these discovered topics, we curate a custom taxonomy of hundreds of topics. With the taxonomy, we developed an ensemble of classifiers to infer the topics of content creators with high-precision (*topical seeds*). The ensemble model aggregates multiple topical signals (features) of a content creators' Instagram account to mitigate the noisy and unreliable features. We showed that we were able to achieve high-precision classification of a content creator $c$, despite the noisy and unreliable features, by aggregating the inferred topics of all content creators $u$ follows. This is a central hypothesis of this work: *a content creator $c$ of topic $t$ follows majority content creators of topic $t$.* The *topical seeds* are then used to spread topics across the content creator graph using a custom designed label spreading algorithm. Our algorithm is designed using various novel observations made about the nature of relationships between content creators on Instagram. The spreading of topics solve the issue of low recall of a high-precision classifier, resulting in a fully topically labelled content creator graph.

We show experimentally that our architecture performs extremely well over a hand-curated dataset of labelled content creators. The architecture presented in this work is deployed in production and used to achieve high-performing social media advertising campaigns. The architecture is continuously evaluated in real-time through a user-facing topical content creator search engine.

# Acknowledgements

I understand my success to be the a combination of my own ability to reason and the people I expose myself to: "none of us are as smart as all of us"[1]. To this end, I'd like to express gratitude for a few external entities:

- The deeply inspiring research literature which provided a clear pathway to walk along.
- To Immanuel Kant, who through his essay "Answering the Question: What Is Enlightenment?" (1784), provided me with the guiding phrase "Sapere Aude" - "Dare to be wise" and its accompanying insight: the courage to act on my own insights and understanding. This idea grounded me in those times where the challenges seemed unsolvable.
- My compadre Thevindu Edirisinghe, who was always there to discuss and contribute to the solutions of complex problems, which in turn stimulated ideas in this work.
- My colleagues at Filli Studios, for their support both emotionally and intellectually, providing assistance in the tasks which required expertise and experience in the space of Instagram. In particular, Nia Pickering, for her exceptional insight in the topical structure of Instagram and her continuous effort in building the taxonomy.
- My supervisor Alex Carver, for his continuous support through this project.
- Finally, my family. For without them, none of this could have been possible.

[1]- Kenneth H. Blanchard

# Contents

# 1 Introduction

## 1.1 Motivation

### 1.1.1 Crowd Driven Innovation on Instagram

**Instagram's Unique Vocabulary for Visual Expression.** Instagram is a visual OSN with 800 million users [37] on which 95 million photos are uploaded each day [2]. In addition to Instagram's scale, it provides a unique vocabulary, which empowers users to inter-inspire and communicate complex visual ideas in ways which were not possible before. Utilising this new visual form of expression [20], youth are now building a digital self at the same pace as their physical and emotional ones.

**Inspiration and Creative Growth.** Consequently, users have been enabled to connect at scale, specifically, forming (tight) topical hubs of extremely talented content creators[1]. These content creators have now been able to inter inspire each other, leading to the formation of new creative fields. Furthermore, high-quality content is being produced on a **diverse range of topics**, ranging from fashion illustrators (Sub-Figure 1.1b), to artists who paint directly onto humans (Sub-Figure 1.1c) to cultures around moving the human body (Sub-Figure 1.1a). The formation of these topics is a key motivator for this work, which provides us with a unique means to study how people interact, communicate and collaborate over these topics at scale.

### 1.1.2 Improvements to the Platform

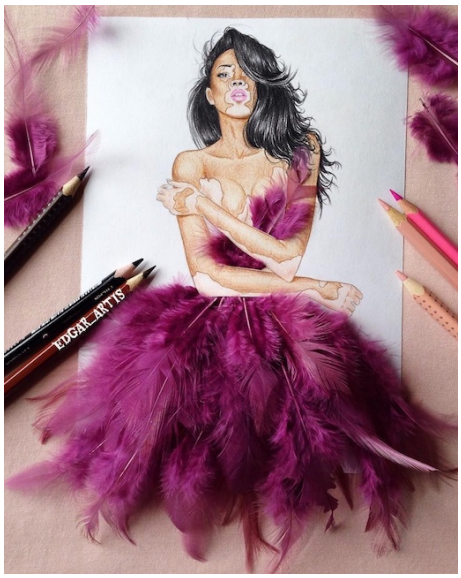
At this point, we have discussed the existence of a diverse array of topical user generated content[2] (UGC) on Instagram. We now continue to present a means to utilise these insights to create various real-world applications. To begin, we discuss applications for business, then continue into users and content creators motivations.

**Discovering and Reaching a Wider Audience.** Looking at the platform in terms of achieving business goals, brands and content creators aim to grow their audiences to new relevant users. Intuitively, the most effective means to achieve this is through *collaborations* and *word of mouth recommendations*. For instance, a Yoga content creator may create a series of Instagram posts with a Yoga brand. In these

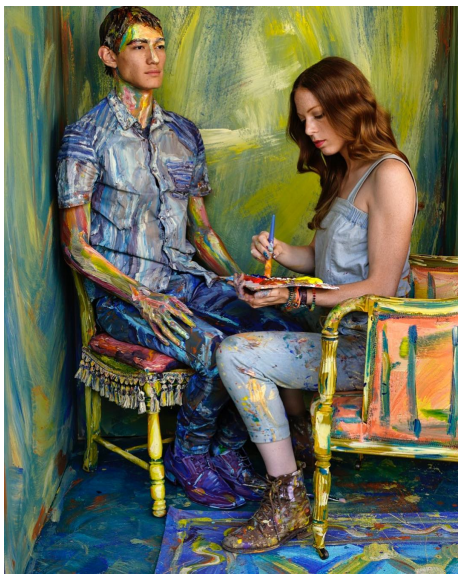
[1] Content Creator of topic $t$: Someone who produces high quality content predominantly of topic $t$.

[2] Content generated by users on social media platforms [78]

(a) @portal.ido - A culture around movement




(b) @edgar_artis - A fashion illustrator



(c) @alexameadeart - A artist who literally paints human beings

Figure 1.1: A sample of UGC from three different topical content creators on Instagram

posts, the content creator may promote brand, thereby exposing the brand to the Yoga content creators *highly-relevant* audience. In order to achieve this, users would need to attempt to navigate the (currently) un-organised platform to find relevant collaborators. A brand/content creator may want to search for content creators to collaborate with, or may want an application to automatically recommend content creators to them with similar content to their own and their audience interests. Hence, this navigation of topical Instagram content creators can manifest itself in two forms: *search* or *recommendations*.

**The Search for Content Creators in Niche Topics.** At the present moment, there are no features on the platform (or elsewhere) that enable users to find content creators by topic, for instance to discover *Miniature Art* content creators. At the writing of this dissertation, Instagram provides a means to search by *hashtag*, whereby a searcher is presented with the posts that include the hashtag. To

demonstrate this, the results from searching the hashtag *#miniatureart* on Instagram includes posts by people who (i) produce content of miniature art, (ii) may have only made one post about miniature art (an art gallery for instance) or (iii) a user who is not relevant to miniature art and solely aims to become popular by making themselves more discoverable by other users. Due to (ii) and (iii), searching via hashtags is a unreliable/noisy means to discover topical content creators. Additionally we note that hashtag search provides results for posts, not topical content creators.

**Recommendations.** Instagram offers "suggestions" when viewing the profile of a user, providing recommendations to the user. This often provides unsatisfactory results for the purpose of finding similar content creators, which can be demonstrated in Figure 1.2. In this scenario, the user is a painter who paints directly on the human body (Figure 1.1c), so one would desire to be recommended similar painters. Figure 1.2 shows that Instagram has recommended two *painters*, an *actress*, a *nature/world photographer*, a *contemporary art museum*, an *art gallery* and *street art community page* as the top 8 suggested accounts. From the perspective of finding similar painters, Instagram's similar feature has under-performed. From our observations, and Instagram's paper on *Discovering Topical Authorities* [54], we believe the suggestions feature has two prominent flaws that make it unsuitable for discovering content creators. These are:

*Popularity Bias* — It is somewhat biased by popularity, providing results including popular topical authorities (e.g. National Geographic and art galleries). We aim to not only discover topical "authorities" but those creators that are not well known but still produce relevant, high quality topical content. The discussion on *authorities* is continued in Section 4.4.2.1.

*User Attributes Bias* — The algorithm provides recommendations biased also by user attributes (e.g. location and previous behaviour), as opposed to solely topical similarity. This is effective for the purpose of personalised recommendations on a social network, but not for topical content creator discovery. We propose a recommendation system similar to Pinterest's recently published paper on their recommendation system *Pixie* [19] and work by Singh and Surendran [71], which performed *biased random walks*. In our work we perform *topically biased* random walks on the content-creator graph.

**Proposed Solution.** In this work we propose a scalable, novel architecture to discover niche content-creators and automatically classify them into a custom-curated taxonomy of hundreds of topics with high-precision (>90%). This architecture is evaluated rigorously against ground truth datasets and user feedback and used to implement a variety of user and business applications. More specifically, it is used in search engines, recommender systems and advertising campaigns. The details of the applications are discussed in more detail below and in Chapter 7. Our practical aim for this work is to provide a unique means to navigate the topical world of Instagram. For this practical aim to be achieved we implement and deploy our architecture into

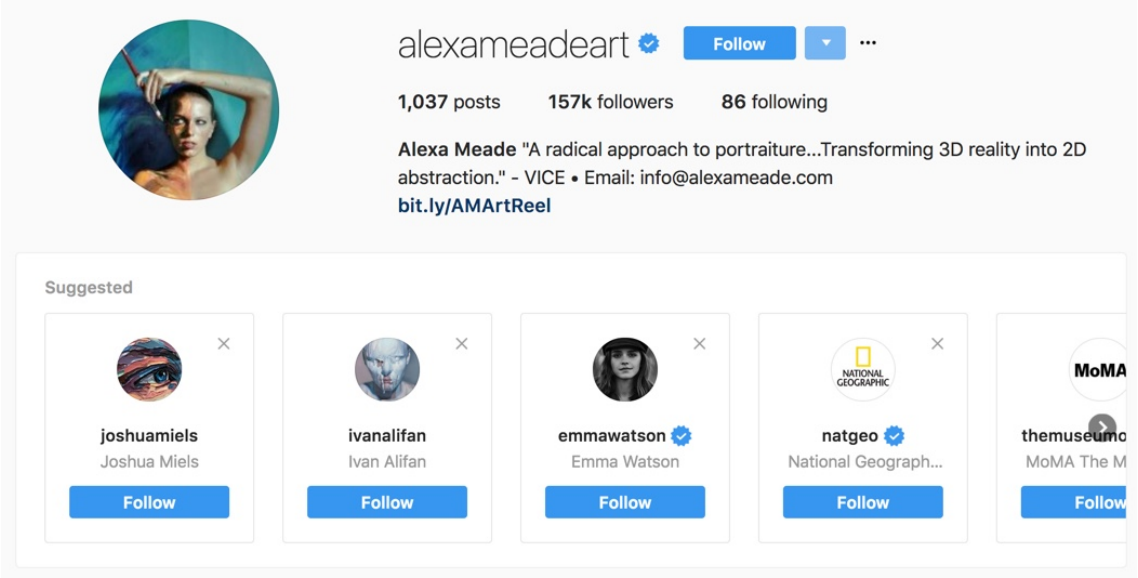



Figure 1.2: Instagram's suggested users based on @alexameadeart

production, where it is applied to a dataset of millions of content-creators. These real-world applications provide us a means to continuously and iteratively improve our architecture based on feedback collected from end-users. For evaluation of our user-facing applications see Chapter 7.

**User Interest Modelling and Recommendations.** Utilising our architecture, we implement user-interest modelling and recommendation systems. User interests modelling is the process of determining the topical interests of a user (Section 2.2. More formally, it can be described as the function $\mathcal{I}$:

$$\mathcal{I}(U) = \{t \in \mathcal{T} \mid InterestedIn(U, t)\}$$

where $\mathcal{T}$ represents the set of topics described by our taxonomy. A user's interest are predicted by performing inferences based on the content creators they follow. This is based on the hypothesis that a user follows a content-creator of topic $t$ then the user is likely to be interested in $t$. The predicted user interest can be used to make further recommendations to the user, showing them undiscovered content-creators that they are likely to be interested in. Eksombatchai et al. [19] have shown experimentally that user engagement on Pinterest was improved up to 50% after improving their recommendation algorithm. This was due to recommending previously undiscovered content. We believe the same also holds on Instagram where a large number of exceptionally talented content creators remain undiscovered. Additionally, to further emphasise the importance of a new form of recommendations, we highlight the recent and inspiring findings of Stoica et al. [75], that Instagram's recommendation algorithm exacerbates "the under-representation of certain demographic groups at the top of the social hierarchy". Specifically, through rigorous analysis of mathematical models and a recent Instagram dataset, they found "an algorithmic glass ceiling that exhibits all properties of the metaphorical social barrier that hinders groups like women or people of colour from attaining equal representations".

**Targeted Advertisement.** From a purely business standpoint, this architecture

can be used for highly-effective targeted advertising, with the aim of maximising return-on-investment of campaigns by advertising only in the places where the viewers are most likely to be interested in the advertisement.

## 1.2 Key Challenges and Solutions

As discussed above, the nature of Instagram's visual content along with the large number of active users has enabled a range of content creators to form across a diversity of fields. The high level aim is to discover topical content creators and classify them with high-precision into their main topics of content. In this section we describe the most prominent challenges we face to achieve this goal. Additionally we present reasoning as to why previous work would not perform in this setting.

## 1.3 Relevant Taxonomy Construction

In order for this architecture to be used by end-users, content-creators must be classified into a set of unambiguous and interpretable topics. It is important to note that the fine-grained topics that exist on Instagram are not previously known, hence there does not exist a pre-defined taxonomy suitable for Instagram. Taxonomies used by other applications and businesses, e.g. IAB Taxonomy[3] are not suitable for this work as they do not capture the interesting topics that exist on Instagram, and contain many irrelevant categories (e.g. "Personal Loans"). Additionally, Lee et al. [41] observed that popular user interest topics differ amongst Instagram, Twitter and Tumblr - for instance a user may share posts about fashion on Instagram and politics on Twitter. This provides a unique challenge of creating a taxonomy that captures as many niche topics on Instagram as possible. To solve this challenge, a large proportion of past work has used a topic modelling approach. This approach implements a statistical model to discover abstract "topics" that occur in a collection of documents. Topic modelling groups related keywords together to form topics, these groups then must be interpreted and labelled. For instance, it may form a group called `Topic08` with keywords {"*Cat*", "*Dog*", "*Horses*"}, which could then be manually labelled as "*Animals*". A human annotator would have to attempt to deduce a topic label from a list of words with their associated probabilities. This results a large cognitive overload and consequently, noisy topic labels. Additionally, topic modelling approaches often merge topics together, which is undesirable in our use case as we aim to capture niche topics. To exemplify this, topic modelling may merge the topics "*Bohemian Fashion*" and "*Bohemian Decor*" due to the similar vocabulary with which they are discussed. Through experimentation we found that state-of-the-art topic models were only able to discover high-level topics like "Photography" and "Travel" in our sparse and noisy text. These drawbacks renders these methods used by previous research unsatisfactory for our work.

[3] https://www.iab.com/guidelines/taxonomy/

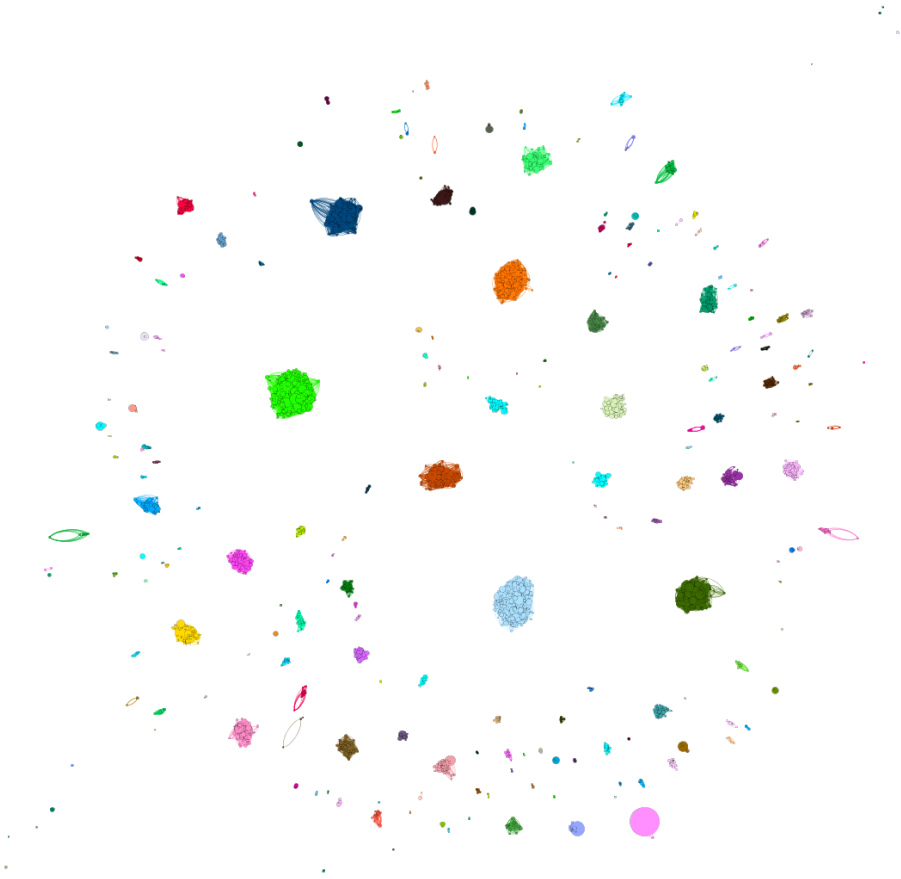


Figure 1.3: A visualisation of the tightly knit topical communities of content creators (nodes) generated by our seed expansion community detection algorithm. An edge between nodes indicates the following relation and a community is labelled by a colour.

**Our Solution.** In order to combat the non-trivial problem of discovering and organising the vastness of topics that exist on Instagram, we undertook an approach which combined manual curation along with exploratory approaches. Specifically we started by utilising existing taxonomies and knowledge bases to provide the groundwork of a general taxonomy. Next, to enrich our taxonomy with the niche and hidden topics that on Instagram we undertook a community detection approach. The benefit of community detection algorithms in this setting is that the connections between users provides much richer information than the text on profiles. Specifically, this involved devising a novel seed expansion algorithm utilising several hypotheses we made about our dataset to form highly-connected communities of topically coherent content-creators. For instance Figure 1.3 shows a visualisation of one of our generated graphs. The communities detected in the graph shown include niche topics such as *Drag Queens*, *Rabbits*, *Minimalist Lifestyle* and *Modern Bohemian Fashion*. Each of these communities is then used to discover hidden topics on Instagram to enrich our taxonomy. Finally, to ensure that subjectivity did not bias the curation of this taxonomy, we collaborated with a team of Instagram experts at Filli Studios. The details of our approaches and results are presented in Chapter 3.

### 1.3.1 Accurate Reasoning in a Highly Deceptive and Noisy World

**Poor Topical Signals**[4]**.** High-precision content classification faces unique challenges because unlike ordinary documents, textual features on Instagram are created informally by biased humans, which renders them sparse, noisy and unreliable. Sparsity of textual features refers to the fact that large portion of users often have empty biographies and short post captions. Conversely, those users who have non-sparse textual features are frequently unreliable and noisy. For instance, as Pal et al. [54] noted, it can be easily observed that users often create non-topical posts like their social lives. It is also common that users will discuss topics that they are interested in, but are not content creators of e.g. one may write in their biography "Wine lover", but the topic of her content is about her poems. In conclusion, traditional text-based methods and topical modelling (Section 2.2) cannot be solely employed for this work. In Section 4.3.1 we take an in depth analysis of various topical signals of an Instagram profile, including hashtags, emojis and hyperlinks to external websites and how we can utilise them to determine the topical content of a content creator with high-precision.

**Non-Topical Users and Self-Presentation on OSNs.** A prominent issue of classifying content creators lies in the abundance of non-topical and passive users. If we were to adopt previous approaches that only use the text on a profile, then we would achieve low-precision topic identification due to large amounts of noise. This noise is predominantly caused due to normal and non-topical users, who are sometimes confused with content creators in our dataset. The problem we observe with normal users stems from a phenomenon called *self-presentation*, where people exhibit ideal self-image through their posts and hence their profile features (text) may indicate that they are topical content creators, but in truth, they are not. This noise would have major negative impact on downstream applications, where it is of importance to find high-quality content creators in specific topical fields. The idea of self-presentation is presented by Goffman [28], where he concludes that people are actors on a "social stage" who actively create an impression of themselves. In this social world, we put on a "front" in order to project a certain image of ourselves (call this part of our "social identity"). Impression management involves projecting an "idealised image" of ourselves. This has also been observed literature, where Shakespeare wrote that "All the world's a stage"[5].

**Conclusions and Solutions.** To account for the noise at an individual content creator's level, we undertake an aggregation approach. Decision Aggregation (DA) in combination with strong hypotheses about the social connections of content creators and users is a central theme of this work. Some examples include: *aggregating user features to discover topics in topical communities*; *majority weighted voting to classify biographies and websites*; *hard voting ensemble on multiple user features to achieve high-precision* and *aggregated, biased random walks to model a users interests*. Instead of using the text in a biography of a user $u$ as the only topical signal, one could aggregate all the biographies of the *content creators* that $u$ follows, provid-

[4] An indicator of a user producing content on about§ topic

[5] As You Like It, Act II, Scene VII [All the world's a stage]. William Shakespeare, 1564 - 1616

ing a more robust topical signal. This solution may mitigate the *Poet - "Wine lover"* problem demonstrated above. This method of aggregation is based on the hypothesis that *the poet is likely to follow other poets*. Previous literature has shown that similar approaches are highly effective in settings that involve modelling user interests based on noisy features [88], dealing with noisy experts [66] and heterogeneous data [18].

## 1.4 End-User Requirements

Information Retrieval systems are only useful if the data has clear, accurate and detailed organisational structure. Additionally, the labelling system should have high coverage[6]. This leads to two important requirements we set out to achieve:

- **High-Precision.** The architecture should only label a content creator $u$ with a topic $t$ if it is certain that $u$ produces content of topic $t$.
- **High-Coverage.** To enable users to discover high-quality content creators, we must aim to have high coverage, having labels which are likely to cover as many types of content creators as possible.
- **High-Recall.** Labelling as many content creators with niche topics as possible. Recall measures how well a model finds all relevant cases in a dataset.

The details of these metrics and how we used them is described in detail in Section 4.5.2. The requirements of high-precision, high-recall and high-coverage make this work particularly difficult. The first requirement is typically satisfied with an abstaining classifier, only returning a prediction if the classifier is sufficiently confident. High-recall can be achieved through sacrificing some precision. We solve this challenge with a multi-component architecture that utilises specific domain-relevant information and hypotheses we take about users on Instagram. With our hypotheses and insight about the platform, we customise state-of-the-art algorithms to build highly-refined components that attempt to maximise their own metrics. These components are then combined into an architecture that aims to achieve the maximal combination of the metrics. Specifically, we first curate a custom taxonomy to capture the fine grain topics on Instagram to achieve high-coverage of topics, that is used by a pipeline that labels content creators with topics with high-precision and very little supervision.

**Achieving High-Recall.** This small labelled dataset is used to classify the remainder of content creators who have sparse and unreliable textual features. To achieve this, we propose a novel semi-supervised learning (GB-SSL) approach, whereby topics are smoothly spread from labelled content creators across the follower graph. State of the art topical authority discovery [54] shows the success utilising of local and global assumptions about graph connections between users. In their work, in addition to the rest of the research around GB-SSL on OSNs they utilise the idea of *homophily*[7] to apply these algorithms to graphs of *users*. We further this approach

[6]The proportion of the data which is labelled

[7]Homophily is the property also known as "birds of a feather ock together", which describes that people who share similar attributes group together.

by making a stronger assumption about connections between users: by only considering *content creators*. Using variations of Hypothesis 1, we spread labels across the graph of content creators. Chapter 5 provides details of our implementation and evaluation against competing supervised and semi-supervised algorithms. After using our GB-SSL algorithms, we achieve *high-recall*, *high-precision* and *high-coverage*. Initial evidence of the validity of Hypothesis 1 can be intuitively observed in Figure 1.3, where tightly connected topical communities form when content creators are grouped by social connections.

**Hypothesis 1.** *If two content creators are connected on Instagram then they are likely to produce content of similar topic.*

**The Impossibility of Training Data Acquisition.** It is important to note that acquiring high-quality training data over a large taxonomy for the classifiers described above is a non-trivial task. Typically, past work has utilised crowd-sourced human annotators, for instance Amazon Turk[8]. Due to the size of our taxonomy and the cognitive overload it would impose on the human annotators and other issues discussed in Section 4.2.1, this approach is ill-suited to our work. Additionally, the platform and previous work provides no means of providing topically labelled data for Instagram which is relevant to our work. We discuss how we tackle this problem in Chapter 4.

**Large Taxonomy.** Lastly, building a multi-label classifier over a large taxonomy of related labels provides many unique challenges. As highlighted by [26], these challenges are due to exponential growth of combinations of labels to take into account and also to the computational cost of building and querying the models. In addition, multi-label data usually present features such as high dimensional, unbalanced data, and dependencies between labels. See Chapter 4 to see the details of how we solved this challenge of multi-labelled topic classification.

## 1.5 Contributions

**Overview of Our Work.** The first component of our architecture is used to assist in curating a growing taxonomy of hundreds of topics covering both niche and high level topics that are highly relevant to user generated content (UGC) on Instagram through topical community detection algorithms (Chapter 3). Next we determine labels of some content creators (*seeds*) for each topic with high-precision through an automatic seed detection and classification pipeline (Chapter 4). With the seed content creators, and Hypothesis 1, the labels are spread across a graph of content creators resulting in a fully labelled graph of high precision, recall and coverage (Chapter 5). An overview of this architecture is visually presented in Figure 1.4. Finally, we instantiate this architecture in the form of recommender systems and user topic modelling applications through random walks on a bipartite graph connecting the content-creator and user graphs (Chapter 7).

To summarise, we list our contributions below, classified into three categories: Research (**R**), Business/Societal (**B**) and Engineering (**E**) Contributions. Each begins with their unique identifier.

[8] https://www.mturk.com/

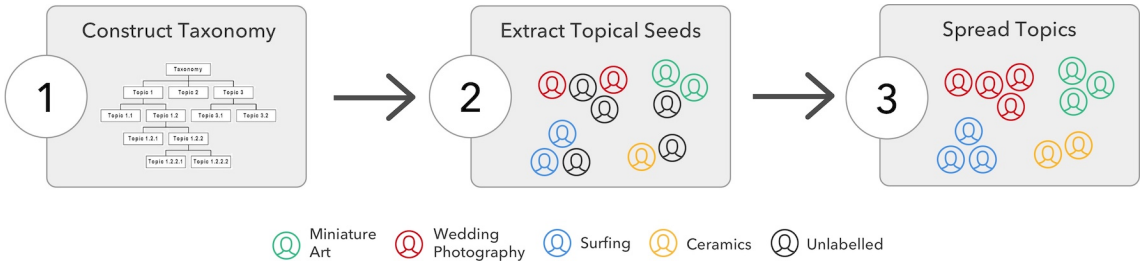



Figure 1.4: A visual overview of the architecture presented and developed in this work. Components 1, 2 and 3 refer to Chapters 3, 4 and 5 respectively.

- `TAX` **B, R, E** A first attempt to constructing a taxonomy which captures a large portion of topical content on Instagram
- `T-LCD` **R** A novel algorithm topical local community detection algorithm for niche topical communities of content creators
- `CLF` **R** A first attempt to classify content creators on Instagram, with high-precision into a large taxonomy of potential topics
- `LBL-DATA` **E** A robust pipeline to automatically label content creators of high-precision topics with minimal supervision
- `TS-LS` **R** A novel modification of the label spreading algorithm that spreads topical labels across a graph of content creators
- `ARCH` **E** A scalable architecture to infer the topics of Instagram content creators, deployed in production and used in multiple real-world applications
- `SEARCH` **B, E** Develop and deploy a search engine allowing users to navigate the topical content on Instagram, specifically providing the first content creator search engine with niche topics in a variety of fields
- `RECOM` **B, E** A topical content creator recommendation system, allowing users and brands to find new high-quality content creators

# 2 Background

This work presents a means to solve the problem of high-precision topic classification of Instagram content creators across a large custom-curated taxonomy. To achieve this we present the research and development of architecture which is comprised of many independently developed components. Each component aims to solve a problem in a unique and mostly independent way. Specifically, we draw inspiration from previous work in range of fields and provide novel modifications, insights and approaches to solve the core objectives of this work. This leads to the challenge of concisely presenting the related work of a wide array of fields.

**Independently Modular Structure.** Furthermore, we aim to present our work in a way that allows the user to draw inspiration and insight from an independent component of our architecture, without being encumbered by related work from other components. For instance, imagine a reader's aim were to understand how to utilise graph-based semi supervised learning to classify UGC into niche topics with small amount of labelled data. In this scenario they would be directed to Chapter 5 without having to traverse the content of other chapters or a lengthy and unrelated background section. Table 2.1 presents an overview the fields explored in each component.

Table 2.1: Fields covered in this work.

| Component | Fields |
|---|---|
| Taxonomy Construction | Hidden Topical Community Detection on OSNs |
| High Precision Topic Inference | Topical Classification of UGC, User Interest Inference/Modelling, Website Classification, Semi-Supervised Learning |
| Label Spreading | Graph Based Semi-Supervised Learning |
| User and Business Applications | Recommendation Systems, Search Engines |

To ensure readers have the ability to utilise this work in a independent and modular manner, we aim to only provide a holistic background in this section. We include specialised and relevant background and related work in each chapter i.e when we begin to solve a new challenge.

This purpose chapter is to provide to the reader: (i) an understanding where this work sits in the current research, (ii) how we address current challenges in a unique and novel way, finally (iii) how each component of our architecture is inspired by previous work. To achieve this we provide a concise overview of the journey of *public incremental advances* in the fields most related to ours.

## 2.1 The Data

**Limit Collection to Content Creators.** This work focuses on collecting only content creators as passive users generally do not have a main topic of content. To be do so we define Algorithm 1 which attempts to use domain relevant measures like *engagement* and *number of followers*. We calculate a users approximate engagement with Equation 3.25. The algorithm follows from the intuition that the less popular a content creators is the higher their engagement must be. We found that these heuristics generated excellent results. This was decided on based on the hypotheses that *content creators are more likely to be content creators than "normal" passive users*. We chose to filter based on local properties, that are relevant to the platform. The main motivations for this are (i) to limit the computational complexity at this stage and (ii) to utilise features directly relevant to our platform. Note that it is not a high-priority to eliminate false-positives at this stage, only to filter out a proportion of users who are likely not to be content creators.

**Algorithm 1** Content Creator Filterer

**Input:** User $U$'s properties: number of followers $F_{in}$ and engagement $\mathcal{E}$
**Output:** Boolean: Whether the user is likely to be a content creator

$S_{xxl} \leftarrow F_{in} \geq 200000$
$S_{xl} \leftarrow F_{in} \geq 100000$ **and** $\mathcal{E} \geq 0.005$
$S_{l} \leftarrow F_{in} \geq 50000$ **and** $\mathcal{E} \geq 0.01$
$S_{m} \leftarrow F_{in} \geq 20000$ **and** $\mathcal{E} \geq 0.02$
$S_{s} \leftarrow F_{in} \geq 10000$ **and** $\mathcal{E} \geq 0.04$
$S_{xs} \leftarrow F_{in} \geq 5000$ **and** $\mathcal{E} \geq 0.05$
$S_{xxs} \leftarrow F_{in} \geq 2500$ **and** $\mathcal{E} \geq 0.10$
**return** $S_{xxl}$ **or** $S_{xl}$ **or** $S_{l}$ **or** $S_{m}$ **or** $S_{s}$ **or** $S_{xs}$ **or** $S_{xxs}$

## 2.2 User Interest Identification and Modelling

User modelling is the most related field to topical content classification and is defined by Zhou et al. [91] as "the process of acquiring, extracting and representing the features of user". The process of *user interest inference/detection/prediction* is a subset of user modelling/profiling. Using Zhou et al. [91]'s definition we establish the following:

**Definition 1.** ***User interests inference/modelling (UII)*** is the process of acquiring, extracting and representing the interests of user.

The problem of inferring the interests of users can be defined as determining the function $\mathcal{I}$:

$$\mathcal{I}(U) = \{t \in \mathcal{T} \mid InterestedIn(U, t)\}$$

where $\mathcal{T}$ is a set of possible topics that a user on Instagram may be interested in. User interest inference is particularly similar to our work as it typically sets out to infer a users interests via various topical signals: indications that a user is interested in a topic. In our work we aim to use topical signals to infer the topics of the content produced by a user on Instagram, hence draw inspiration from the field of user interest inference. The key difference between our field of content creator classification and UII is that of the interpretation of topical signals. For instance, we can infer that a user who includes "Wine lover" in her biography is interested in *Wine*, but we cannot infer that the user produces content about wine. This makes our problem particularly challenging in that we must more meticulously utilise topical signals, attempting to eliminate such false positives. This work provides a unique outlook to the field of UII and content creator classification, providing more accurate topical signals by studying the relationships between content creators and other users. In the following sections we describe the various dimensions that have been explored in tackling user interest inference and how they inform this work.

### 2.2.1 Textual Topical Signals

#### 2.2.1.1 Leveraging Knowledge Bases

It is possible to represent a users text and in-turn interests by conceptual nodes and their relationships constructed using pre-existing knowledge bases (KB) for example Wikipedia. A concept can be an entity, category or class from a KB. This approach is referred to as concept-based interests that involves entity extraction and linking. Due wide range of applications of knowledge bases, there has been effort in automatically constructing large scale general-purpose structured knowledge bases that describe concepts, their semantic categories and relationships between them. Popular constructions include DBPedia [1] and Wikidata [2], where on Wikidata, the concept National Football League (NFL) has statements that link it to other concepts in the KB, such as `instance_of:Professional_Sports_League`, `sport:American_Football` and `main_category:National_Football_League` [3].

**Bag of concepts.** A users interests can be represented by a bag of concepts. Concepts that can be directly extracted from a users profile (e.g. in a bio [54] or captions [69]) are commonly called ***primitive interests***. This representation for modelling interest topics has been shown to be more effective than previously proposed representation (e.g. bag of keywords) [44, 58].

**Semantic Enrichment.** In combination with the bag of concepts approach, it is possible to infer `deduced` or `propagated` interests to enrich the set of concepts. For instance, knowing that someones primitive interests are the concepts `dbpedia:The_Wombats` and `dpedia:The_Black_Keys`, we can use the relationships of concepts in DBpedia to augment the interests profile with `dbpedia:Indie_rock` through the ontology

[1] http://wiki.dbpedia.org/
[2] https://www.wikidata.org/wiki/Wikidata:Main_Page
[3] https://www.wikidata.org/wiki/Q1215884

`dbpedia-owl:genre`. The Table 2.2 shows how Piao and Breslin [59] exemplifies this method with a Tweet. DBpedia provides additional information that can be used to enrich the user interest profile, e.g. related entities. Peña et al. [55] use multiple knowledge bases to enrich a user profile by checking if categories from multiple sources are relating to the same concept using Dbpedia's *categorySameAs* property. The authors describe an example whereby they first extract the category "Music", from the URL *soundcloud.com* using OpenDNS[4]. They then use DBpedias *categorySameAs* property to enrich the profile with the new concepts `dbpedia:Streaming` and `dbpedia:Social_networking_services`.

Table 2.2: Manually extracted DBPedia concepts from a sample tweet [59]

| Tweet | Primitive Interests | Deduced/Propagated Interests |
|---|---|---|
| My Top 33 #lastfm Artists: Eagles of Death Metal (14), The BlackKeys (6) and The Wombats (6) #mm | `dbpedia:The_Wombats`, `dbpedia:The_Black_Keys` | `dbpedia:Indie_rock` |

**Bag of Categories.** It is also common to ground interest topics on categories from a pre-existing knowledge base [47, 53, 54, 60]. The authors of [53] used DBpedia to extract concepts from text in the form of categories. In their work they also exploited the structure of the DBpedia category graph to create a discounting strategy into the relevance of an extracted category to the user's interests. [54] used expert curator within the OSN studied to convert a large list of top-level Wikipedia categories into a subset of whitelisted categories relevant to their problem. In [60], the standard model of bag of categories is extended to use both categories and entities as a interests modelling strategy. For instance they would extract the entity `dbpedia:Apple_Inc` and category `dbpedia:Category:Electronics_companies`. Furthermore within the context of Google+ link recommendation, the authors showed that category and entity based performs better than just entity based.

**Discussions.** In conclusion, concepts and categories from a predefined KB have been shown to provide an effectively represent user interest, and provide useful features to enrich the user interest profile further. To consider this approach we highlight an important bottleneck within this approach: the ability to accurately extract concepts from text on Instagram. It is not the case in practice that the exacted concepts are as "clean" as Table 2.2, for instance Table 2.3 shows the concepts extracted from an *House Plants* content creator's biography using DBpedia Spotlight entity extraction and linking API [15]. In the example in Table 2.3, the inferred interests in the next stage. In order to use this approach as a stepping stone to classify the topical content of curators we require the linking to be extremely accurate. In conclusion, the approaches highlighted in this section mostly

[4]A domain tagging service

extract concepts from a profile, enrich them then use them as topics to represent a users interests. From experiments we have found that using this approach does not meet our core requirement of high-precision. For this reason, we draw inspiration from the work presented in this section to improve the accuracy of our labelled data extraction pipeline in Chapter 4 and to improve the quality communities generated by our topical community detection algorithm in Section 3.2.

Table 2.3: Noisy Concept Extraction.

| Bio | DBPedia Concepts |
|---|---|
| Beyond Sunflowers Greening The Great Indoors - For all your urban jungle and houseplant needs. | *Helianthus*, *The_Great_Indoors_(TV_series)*, *Jungle*, *Houseplant* |

#### 2.2.1.2 Topic Models

One of the most common approaches employed within interest inference in OSNs is an unsupervised one. It is often employed when topics are unknown: the *cold-start* problem where topic vectors need to be created. These methods learn topics from a (often large) set of text documents. Topic modelling is a statistical model that uses word co-occurrence in documents to cluster words into related topics. For instance Latent Dirichlet Allocation (LDA), a statistical generative method that discovers topics and represents them by word probabilities. This method can use the words in new documents to infer topics, generating a probability distribution over topics. Pu et al. [62] combined methods described in Section 2.2.1.1, whereby they found Wikipedia categories for keywords and key-phrases in topics extracted from LDA. They found this method improved on traditional topic modelling approaches, specifically it produced more accurate, meaningful and coherent interest topics. This further validates the benefit of utilising knowledge bases in topic extraction.

A variation of LDA, namely Labelled Latent Dirichlet Allocation (L-LDA) is employed by [6] that constricts topic models in the same was as LDA, but in a supervised manner. L-LDA is supervised by providing each document with a label, constraining the topic model to use only those topics that correspond to a document's label set. Their method included running L-LDA on tweets selecting 300 of the most common hashtags from tweets to represent topic labels. Through this method, they discovered 300 topics, each containing the most frequent ($\approx 50$) words in that topic. Once determining topics, they defined categories from Pinterest category graph and attempted to map topics to these top level categories by using AMT[5], whereby each Turker would assign categories to each generated topic. Quercia et al. [63] use L-LDA to classify the topic Tweets with the aim of user interest inference. They showed that L-LDA outperforms a SVM classifier with the task of Twitter profile classification. Despite the improvement, L-LDA suffers from the same issues as standard LDA from subjective labelling other problems of merging

[5] https://www.mturk.com/

of similar topics, as described below.

*Merging of Topics.* Similar topics are often merged together, e.g. nature photography and landscape photography. This is unideal for inferring niche interests over our taxonomy.

*Ambiguous Labels.* The output of topic models is a list of groups of words with arbitrary titles (e.g. topic01, topic02.. etc). It is down to the user to manually label these groups. Raw topics clusters are highly un-interpretable, and hence can easily be poorly subjectively labelled. For instance, the topic modelling approach may form a group called `Topic08` with keywords of high probability = {"*Cat*", "*Dog*", "*Horses*"}, that could then be manually labelled as "*Animals*". These topics are hard to interpret and hence are difficult to manually label unambiguously, e.g. `Topic08` could have also been subjectively interpreted as "*Pets*". Due to the vastness of topics on Instagram and the problems described above, it would result in large error when labelling topics that would propagate errors to later stages of my architecture. Additionally, our end-to-end system would be directly used by users, that makes it of a large importance to have topic labels that are easy to interpret and accurate.

**Discussion.** Xin ZHAO et al. [85] state that the short nature of Tweets limit the performance of standard LDA, we expect a similar observation holds for short Instagram text. Finally, Mehrotra et al. [46] describe that implementations of LDA applied to Twitter content produces mostly incoherent topics. Due to the short and informal nature of text on Instagram latent topic modelling techniques such as LDA will have significantly worse performance than when applied to longer and more semantically rich documents like blog posts and news articles. There has been work that has aimed to acquire higher quality data that can be associated with Tweets to then be fed into an LDA model. For instance recent work by Nigam et al. [52] describes a topic identification pipeline that extracts nouns from tweets then feeds them into the Google search engine, building larger text documents for that they performed LDA with. Inspired by this and other similar work, we used a similar aggregation approach whereby we use the website linked to an Instagram to acquire higher quality data. In conclusion, the drawbacks described LDA unsuitable make it unsuitable for this work.

#### 2.2.1.3 Dictionary Approaches

Spasojevic et al. [73] address some of the issues discussed above with the quality of text on OSNs by using curated dictionaries and whitelisted mappings, converting the text on a profile into high-quality topics. Specifically, they first extract n-grams from the text on a users profile and map them to a bag-of-phrases using an internal dictionary mapping n-grams to phrases. Next they map the bag-of-phrases to a bag-of-topics based on exact match and rule based synonyms.

**Discussion.** This method provides high-precision topic extraction with minimal false-positives, however requires a large amount of human labour to curate high-quality `n-gram -> phrase` and `phrase -> topic` maps. For this reason, and to promote scalability, we only use hand-curated maps in those components that require high-precision. Specifically, we adopt an approach inspired Co Training [8], whereby one of our classifiers can be used (with some supervision) to create `keyword -> topic` maps (another classifier). Our methods are described in Section 4.4.

### 2.2.2 Leveraging Internal Structures

Often OSNs include organisational structures whereby users can group together content or users. These structures can provide means to (i) identify and represent relevant interest topics for the OSN and (ii) acquire pre-labelled data. Utilising such structures can solve some of the issues that stem from bag-of-words approaches.

**Twitter Lists.** A common structrue leaveraged for user interest modelling on Twitter is Twitter Lists, where users can curate a List of Twitter accounts. When creating a List, a user specifies a name and an optional description. For example the account @BarackObama could be added to the Lists "Politics", "Celebs" and "Government" created by various users. Further, users can subscribe to many Lists created by themselves or others. Utilizing this information, [3, 25] define interests by the names of lists that users create. Lists can continue to be used to infer the interests of an expert[6] user $u$ by extracting the frequent words that appear in the names and descriptions of Lists they appear in. The topics of @linuuxfondation can be inferred by this technique as "Tech", "Linux" and "Software".

**Facebook and Pinterest.** Outside of Twitter, Cinar et al. [12] use the predefined Pinterest Category Graph and Jiamthapthaksin and Aung [36] are able to define user interest topics through Facebook Pages' tagged categories. For example Cinar et al. [12] represents a users interests as a frequency distribution over the categories the users pins are tagged as.

**Discussion.** Unlike Twitter, Facebook and Pinterest, Instagram provides no public information about a users likes (defined by topics) and only has functionally to search by hashtags or usernames. Employing a method that utilises the hashtag search feature on Instagram was proposed by Ferrara et al. [22] to model user's interest topics. This approach involves searching Instagram for a hashtag representing a category, for example `https://instagram.com/explore/tags/football/` lists posts that include the hashtag #football. This method suffers from category cohesion, appropriateness and polysemy. Category cohesion and appropriateness is mostly caused because people often tag photos with a variety of hashtags and don't exclusively produce content in that area. We discuss this issue in more detail in Section 4.2.2 with a detailed illustration of our solutions.

[6] Ghosh et al. [25] defines a user $u$ as an expert within a topic $t$ if and only if $u$ occurs in at least 10 Lists of topic $t$.

## 2.3 Entity Linking and Extraction

### 2.3.1 Preliminaries

Throughout this work we utilise knowledge bases in order to improve the performance of various of our algorithms. In particular, we use Wikipedia articles/categories to ground keywords upon. Many of our components that external knowledge bases require the functionality to find mentions to Wikipedia articles/categories in text. Figure 2.1, illustrates both the process of entity extraction and linking (EEL) and the difficulties provided by the short, noisy and informal nature of Tweets (and similarly Instagram captions). For instance, in Figure 2.1 $t_1$ lacks the context to confidently link "Bulls" to "Chicago Bulls", this problem is often called *entity disambiguation*. Most frequently in the context of *interest analysis* and *topic classification*, research use external APIs are used to perform entity extraction and linking. To demonstrate this, Piao and Breslin [57, 58] use `Aylien API`[7] to extract primitive interests as concepts from documents. However, the rate limits imposed by such APIs are to restrictive to my work, where we aim to have an initial set of content creators of the magnitude $10^6$. Moreover, most external APIs aren't tailored to capture the detail and nuances of concepts that appear on a platform like Instagram.

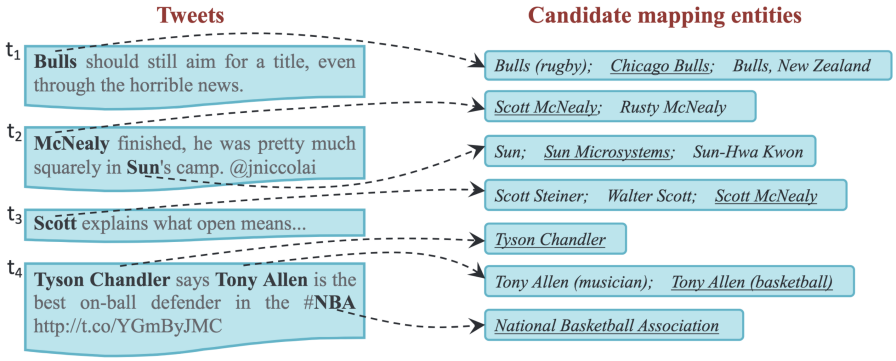



Figure 2.1: An illustration of the tweet entity linking task. Named entity mentions detected in tweets are in bold; candidate mapping entities for each entity mention are ranked by their prior probabilities in decreasing order; true mapping entities are underlined [70].

**Mention Detection.** As a preliminary, we use Guo et al. [31]'s definition of mention detection as "the task of extraction surface form candidates that can link to an entity in the domain of interest". In Figure 2.1 the mention "Bulls" is detected in tweet $t_1$.

### 2.3.2 Domain Specific Linkers

To improve the performance of off-the-shelf entity linkers in specific domains there has been work in combining simple hand-curated rules with the results of multiple entity linkers. Ruiz and Poibeau [68] obtained higher precision and recall through string matching rules.

[7] http://aylien.com

### 2.3.3 Challenges of Entity Linking in OSNs

Guo et al. [31] reason that the performance of entity linking suffers in OSNs (particularly micro-blogs) due to the mention detection part of the process. This problems of poor mention detection and entity disambiguation can be derived from the short, noisy and informal nature text with little context. To address these problems, Guo et al. [31] propose a method of combining both entity linking and mention detection into one end-to-end optimisation problem.

**Discussions.** The two key takeaways here are (i) the nature of text on OSNs makes it difficult to effectively extract mentions and (ii) the short nature of this text provides very little context for an entity linker to disambiguate between different possible candidate mappings. To tackle (i) we aggregate the results from multiple entity linkers (Section 2.3.5). We found this method to be very effective, through direct observation and experiments we observed that DBPedia Spotlight matches more uni-gram keywords whereas Dexter is effective in extracting big-grams, that can be seen in Figure 2.4. In particular, DBPedia Spotlight uses an entity linker that was unable to detect the entity "Vintage Fashion" due to the informal nature of it's mention (all caps). In conclusion, using multiple linkers will assist in increasing the number of concepts successfully extracted. To tackle (ii) there has been work in combining multiple contexts around the text, that include combining multiple tweets on a profile and considering their inferred interests [70]. This approach was shown to been shown to be effective in solving some issues with entity disambiguation, so we experiment with aggregation techniques in Section 4 in attempt to improve performance of entity linking.

Table 2.4: Comparing the concepts extracted and linked by two entity linkers: DBPedia Spotlight and Dexter. In particular, Dexter performs better on bi-grams than DBPedia Spotlight

| Biography | DBPedia Spotlight Results | Dexter Results |
|---|---|---|
| VINTAGE FASHION \| no filter photos, Paris lover | `Vintage_clothing`, `Photographic_filter`, `Paris` | `Photo`, `Paris` |

### 2.3.4 Voting Schemes

Ruiz and Poibeau [68] discuss that entity linking has inconsistent performance over various domains, where different annotators have strengths and weaknesses in each domain. They continue to show how combining the outputs of multiple entity linkers with a voting weighting scheme improves annotation results across four test-corpora.

**Discussions.** We acknowledge that using a voting weighting scheme over the outputs of multiple entity linkers can significantly improve *precision*, but we tackle the problem of precision in a later step with a dictionary approach (Section 4.4). For

this reason we do not apply a weighting scheme to the outputs of our entity linkers to maximise the amount of candidates our entity linker generates.

### 2.3.5 Conclusions

**Prioritisation.** Despite EELs its effect on the performance on various components of our system we limit the amount of attention we spend on improving EELs performance in our domain. Our reasoning for this is as follows: (i) EEL is not the focus of our work and (ii) EEL is a complex problem with many challenges [43].

**Combining Entity Linkers.** In order to improve performance of EEL whilst minimising time spent, we decide to take the approach combining multiple entity linkers. In this work we combine the extracted concepts from the following open source entity linking and extraction frameworks: DBPedia Spotlight [14], Dexter [10], Fast Entity Linker [5] and TagMe API [8]. Our practical implementation of combining these entity linkers is described in Section 4.4.4.

## 2.4 Text Vectorisation

Effective natural language processing is an important pre-requisite of the success of many components of our architecture. Specifically, we use the techniques described in this section to create high-quality vectors from the text on Instagram. Here we provide a brief overview of the techniques we considered in achieving text vectorisation.

### 2.4.1 Bag-of-Words

The most simplest method of vectoring text is by using word frequencies in the form of *bag-of-words*. To generate a bag of words we first define a fixed length vector with the same size of a pre-defined dictionary. Each index in the vector corresponds to a word in the dictionary. To then acquire the vector for a document we would count the number of times a word in the dictionary appears in the document and increment it's corresponding entry in the vector. For example with a dictionary defined as {A, it, and, fashion, portrait, photography, photographer} and we were to vectorise the document "A fashion photographer and portrait photographer", we would generate (1, 0, 1, 1, 1, 0, 2).

**Discussion.** At this point we can motivate the importance of effective text-preprocessing to improve the quality of our text vectors. For instance, by converting text to lower case will prevent the need for storing two entries for each word, and consequently not capturing the relationships between documents effectively, e.g. the documents *A Photographer* and *a photographer* with no text-preprocessing would have completely different bag-of-word vectors. We discuss the text processing methods we employ in Section 2.5. There is one major drawback with a bag-of-words approach: it doesn't capture the semantic meaning of the text by ignoring the context in which words appear. To tackle this we progress to the next approach: *tf-idf*.

[8] https://sobigdata.d4science.org/web/tagme/tagme-help

### 2.4.2 TF-IDF

Term Frequency-inverse Document Frequency (TF-IDF) measures how important important a term (word) is to a document within a collection documents (corpus). To construct a tf-idf vector for a document we construct a fixed length vector in the same way described above, but instead of an index corresponding to the frequency of a word, it is the tf-idf score. We calculate the score for a word $w_i$ for a document $d_j$ as follows:

$$score(w_i, d_j) = tf(w_i, d_j) \cdot idf(w_i) \tag{2.1}$$

where $tf(w_i)$ represents the term frequency of $w_i$: the number of times word $w_i$ occurs in document $d_j$ and $idf(w_i)$ weighting the score with the inverse document frequency: the number of times word $w_i$ occurs in all documents (the corpus). With this equation we can intuitively observe that words with high frequency in many documents will have a low score, while those with high frequency in a specific document but low frequency in all documents will have high relevance (score) for that document.

**Discussion.** TF-IDF makes a major improvement on bag-of-words by considering context. However, this method has limited applications in areas where the corpus changes size and it doesn't consider the relationship between words.

### 2.4.3 Word Embeddings (Word2Vec)

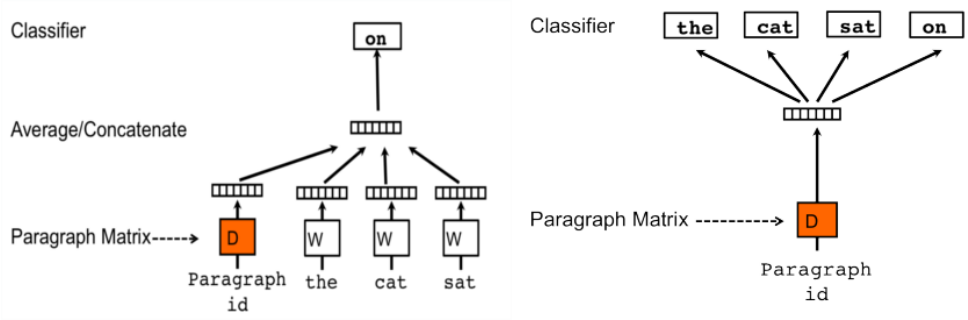
Word Embeddings are dense vector representations of words in low dimensional vector space. They were popularised by Word2Vec [48] where they introduced a new model for distributed words using neural networks. Word vectors produced by these models preserve the semantic similarity between words. For example, the word embeddings learnt from a large amount of data by the Word2Vec model encodes the relationships between the words "king", "man", "woman" and "queen" in the following way:

$$v(king) - v(man) + v(woman) \approx v(queen) \tag{2.2}$$

where $v(w_i)$ is the vector embedding of the word $w_i$. Vectors which preserve these semantic relationships provide high-quality representations of text.

### 2.4.4 Document Embeddings (Doc2Vec)

In our applications we use text which is longer than just a single word. These sequences of words (documents) provide additional context to learn the semantics of words. This motivates the consideration of a means to create *document embeddings*. Le and Mikolov [40] proposed Doc2Vec, which learns document emdbeddings jointly with word embeddings. Doc2Vec was shown to outperform many other methods of generating document vectors [39].




(a) PV-DM training diagram [40] (b) PV-DBOW training diagram [40]

Figure 2.2: Doc2Vec Training Models

Similar to Word2Vec, Doc2Vec learns vector representations of a word by looking at words context, additionally it includes a new vector to the context, which represents the paragraph. Doc2Vec has two models: Distributed Bag of Words (PV-DBOW) and Distributed Memory Model (PV-DM).

**PV-DM vs PV-DBOW.** In PV-DM 2.2a (a), a shallow neural network is trained on a "fake task", which attempts to predict a centre word using context words and a context paragraph. Whereas in PV-DBOW 2.2b (b), the paragraph vectors are obtained by through a training procedure whereby a neural network predicts a probability distribution of words in a paragraph given a randomly-sampled word from the paragraph. The original paper recommends the PV-DM model as being superior.

## 2.5 Text Preprocessing

In order to extract the highest quality vectors it is understood that the raw text should undergo informed text-preprocessing. This section describes some of the approaches we undertook in this work.

**Emojis.** Emojis often encode useful information (usually topical information), for this reason we undergo methods to encode emojis such that they can be used by our text models. Specifically we use the Emoji library[9] which converts emojis into distinct tokens, a form which our text algorithms can use.

**Text Normalisation and Stripping.** In most scenarios we apply the following normalisation and stripping pipeline: (1.) convert text to lower case, (2.) remove emails and urls, (3.) remove punctuation, (4.) remove stop words, and finally (5.) remove tokens smaller than 2 characters in length. It is worth noting that we added a set of custom stop words which we occur frequently in the text in our domain that provide no value.

**Lemmatisation.** Lemmatisation in linguistics is the process of grouping together the inflected forms of (a word) for analysis as a single item [17]. For example, the verb "to walk" may appear as "walk", "walked", "walks", "walking", all of which have

[9] https://pypi.org/project/emoji/

the *lemma* "walk". By reducing words to their lemmas we can intuitively reason that the performance of text models like bigrams (Section 2.5.0.1) and word embeddings will be significantly improved. We utilise the lemmatiser from the SpaCy[10] library. We decide to use a lemmatiser over a stemmer as lemmatisation provides more readable words to assist in interpretability. In the most of our work because we often need to manually explore the results of our text models, which makes lemmatiser a better fit.

**Frequency Filtering.** For building topic models or similarity models we filter out words at extremes, e.g. any words which occur in less than 20 of the documents or those which occur in more than 70% of the documents in the corpus. This helps filter out common words which may negatively impact our results.

#### 2.5.0.1 Bigrams

During this work we found that using bigrams [49] were an effective pre-processing step to improve the quality of text classification tasks and text models like Doc2Vec. This is expected as they intuitively encode more information than unigrams, e.g. the seperate unigrams *Fashion Photography* are less specific than the bigram *Fashion_Photography*.

**Implementation.** We create two separate bigram models using the Gensim library [11], one for the text on websites and the other with text in biographies. We created two bigram models so each was able to capture the relevant semantics for the individual contexts. To create a bigram we combined all the text gathered for the specific context, e.g. all 500k biographies were provided to model. The model learns a score for all pairs of words, essentially learning which pairs of words are more likely to occur together than not. The score for a pair of words is calculated intuitively using the equation below:

$$score(w_i, w_j) = \frac{count(w_i w_j)}{count(w_i) \times count(w_j)} \tag{2.3}$$

where $count(w_i w_j)$ represents the amount of times which the amount of times words $w_i$ and $w_j$ one after the other in all documents and $count(w_i)$ as the number of times word $w_i$ occurs by itself in all documents.

[10] https://spacy.io/api/lemmatizer
[11] https://radimrehurek.com/gensim/models/phrases.html

# Taxonomy Construction 3

## 3.1 Background

A *taxonomy* refers to a hierarchical categorisation in which relatively well-defined classes are nested under broader categories. This creates a formal description and specification of topics. These then provide a standard format to define topics across multiple end-user applications.

**Motivation.** Without a clear and consistent taxonomy of topics, users cannot be categorised into their content types or interest. This categorisation is a prerequisite for our user search and recommendations applications (Chapter 7). In conclusion, the more accurate, low-level and relevant the taxonomy, the higher coverage we our classification algorithms will achieve and in-turn the more effective the end-user applications.

## 3.2 Topical Community Detection

### 3.2.1 Background and Related Work

In this component, we aim to find topical sub-graphs in our network of content creators where the sub-graphs are more densely connected than the rest of the network. For the purpose of interpretability we also aim for the sub-graphs to have little overlap. We will work with the intuitive definition of a community as a cluster of vertices such with more internal edges than those which are external. Additionally, we will make a distinction between *strong* and *weak* communities, whereby a strong community is one where each node has more connections within the community than with the rest of the graph. In a weak community $C$, the sum of all degrees within $C$ is larger than the sum of all degrees toward the rest of the network [64]. Community detection (CD) algorithms can be categorised broadly into two types: *global* and *local*. Global algorithms typically perform better as they have access to the whole graph at once, where they are able to utilise both local and global information. However, due to the size of the content creator graph, huge computational resources are required for global algorithms. As optimality is not a priority in this component, the computational overhead of global CD algorithms becomes unjustifiable. Therefore, I will focus on developing local CD (LCD) algorithm where each community can be grown independently, allowing for a parallel and on-demand sub-graph discovery. In this context, LCD algorithms focus on the sub-graph that

is under study and it's immediate neighbourhood.

**Local Community Detection (LCD).** There has been a large effort in detecting community structure in complex networks [13, 51] using modularity optimisation and minimisation of cuts using spectral methods [82]. Following the adoption of the definition of community we presented above, the most popular algorithms are centred around optimising a quality measure known as "modularity" (and local modularity). Modularity is the measure for strength of a division of a network into communities. Su et al. [76] continue to define that a graph has ambiguous community structure when it contains some weak communities and discuss that modularity based CD algorithms are effective in detecting communities in complex networks with all strong communities but not those with ambiguous structure. Additionally, it is unable to detect small communities as it suffers a resolution limit [27]. It has been shown that most ties in social networks are weak [21] and we believe through direct observation that, like other social network datasets, that our dataset contains predominantly weak communities, i.e. has ambiguous community structure.

**Topical Community Detection.** The works described above focus on using structural properties of graphs and hence generate communities associated with incoherent semantic topics. This makes them unsuitable for discovering topics. In order to overcome this limitation, there has been work in using topic modelling techniques such as LDA [7] or AT [74] with the goal of discovering topical communities. For example Zhou et al. [90] proposes Community-User-Topic, that extends LDA, discovering communities using the semantics of content. These methods that use topic modelling technique are heavily reliant on the quality of textual content in the medium studied. As discussed in length in this work, the text on Instagram is noisy, sparse and unreliable, which renders these methods inappropriate for the goal of discovering topical communities on Instagram. To overcome these issues we provide a local topical community detection algorithm which uses both structural and textual information to discover topical communities on Instagram.

**Hierarchical Community Detection.** Following the hierarchical nature of our taxonomy, we hypothesise that there exists a similar hierarchical structure to the communities which exist on Instagram. For the purpose of niche topic discovery, it remains a priority to develop an algorithm to generate smaller leaf level communities, as opposed to those which represent top-level topics/categories. Practically, if we naively find seeds based on attributes such as node degree (Equation 3.2), we will end up with seeds which will grow into general topics. For instance, a photographer like @nk7 is followed by photographers in a wide range of photographic disciplines and hence has a large degree in our network. Traditional methods of seed selection like [76, 81], greedily pick seeds from the graph in order of decreasing degree. Methods like this will pick users like `@nk7` as seeds, where `@nk7` would be grown into a large, general community of photographers. This can be observed in Figure 3.1 whereby we grew a small number of communities around seed nodes chosen with high-degree. Inspection of these communities visually and analytically indicates that these represent general topics like "Home" and "Fitness". We tackle this issue by devising various quality functions for communities (Sections 3.2.4 and 3.2.5) and for seeds (Section 3.2.6). Specifically, the seed quality functions allow us

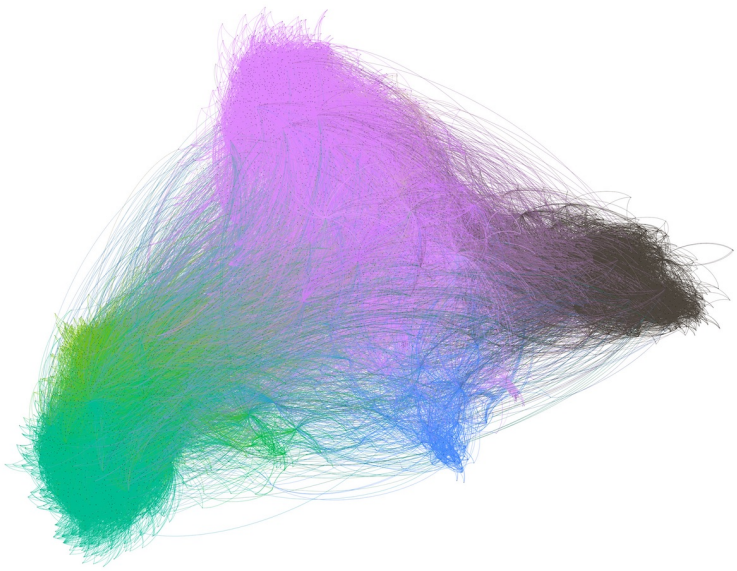


Figure 3.1: The visualisation here represents a graph of content creators, where each colour is a community, a node is a content creator and an edge represents the "following" relation. This is an example of communities which form when poor seeds are selected, i.e. those communities grown from seeds with large degree.

to better assess the "likelihood" of a content creator to be grown into a niche topical community. Content creators are then greedily selected as seeds based on an ordering on the quality functions we define. Consequently, we are able to grow smaller and more niche communities which represent potential leaf nodes in our taxonomy.

**Our Algorithm.** We devise an algorithm that finds important nodes and uses them as seed nodes $\{s_1, s_2, ..., s_k\}$, which are grown into independent clusters $\{C_1, C_2, ..., C_k\}$ using custom defined quality functions. Specifically,

1. A seed set of important content creators are found, using criteria to ensure seeds are well separated and topically independent. We combine traditional definitions of importance with domain relevant statistics (e.g. likes, followers and tags) which guide our seed selection algorithm.

2. Each seed content creator is grown into a community $C$ in parallel by greedily selecting nodes which are in the neighbours of $C$, and adding them to $C$ if it improves the quality of $C$. The nodes are greedily selected using various quality functions we define over a community.

### 3.2.2 Algorithm Setup and Objectives

To guide the construction of the communities and selection of seeds, we define a set of intuitive objectives:

**Criteria 1.** *Communities should have small overlap of users*

**Criteria 2.** *Communities cover as many topics as possible*

**Criteria 3.** *Each community should define a unique topic*

### 3.2.3 Preliminaries

**Setup.** A directed graph $G = (V, E)$ which consists of content creators $V$ and the edges between them $E$. The existence of a edge $(u, v) \in E$ indicates that $u$ follows $v$. We define a cluster $C_i$ as a subset of vertices in $V$, $C_i \subseteq V$. Let $E(C) = \{(u, v) \in E | u, v \in C\}$. In this chapter, we use the terms *cluster* and *community* interchangeably.

**Neighbourhood.** The *neighbourhood* of $u$ represents all nodes that are adjacent to $u$:

$$N(u) = \{v|(u, v) \in E\} \cup \{v|(v, u) \in E\} \tag{3.1}$$

Consequently, the *degree* of a node can be defined as

$$\texttt{deg}(u) = |N(u)| \tag{3.2}$$

which measures the number of connections it has to other nodes.

**Volume.** The *volume* of a cluster $C$ is sum of all degrees in $C$:

$$\texttt{vol}(C) = \sum_{v \in C} \texttt{deg}(v) \tag{3.3}$$

**Bridges.** We define $bridges(C_i, C_j)$ as the set of directed edges between clusters $C_i$ and $C_j$:

$$\texttt{bridges}(C_i, C_j) = \{(u, v)|u \in C_i, v \in C_j, (u, v) \in E\} \tag{3.4}$$

**Cut.** The cut of a $C$ is defined as the number of bridges between $C$ and $V \setminus C$:

$$\texttt{cut}(C) = |\texttt{bridges}(C, V \setminus C)| \tag{3.5}$$

### 3.2.4 What makes a Good Cluster?

In this section we provide a detailed presentation of methods of evaluating the quality for individual clusters, along with practical considerations. Our definitions will be guided by the criteria outlined in Section 3.2.2, ensuring our seeds are grown into niche topical communities.

**Conductance.** Our first quality measure is one which measures the probability that a node $u$ in $C$, after a one-step random walk leaves $C$.

$$\texttt{conductance}(C) = \frac{\texttt{cut}(C)}{min(\texttt{vol}(C), \texttt{vol}(V \setminus C))} \tag{3.6}$$

Equation 3.6 requires computation on the whole graph to calculate $vol(V \setminus C)$. In order to eliminate this computational heavy requirement we assume that the clusters we generate are always much smaller than the whole graph. This allows us to re-formulate conductance as shown below:

$$\texttt{conductance}(C) = \frac{\texttt{cut}(C)}{\texttt{vol}(C)} \tag{3.7}$$

Intuitively, we aim to build community where the probability for a node to leave the community after a one-step random walk is minimum, hence minimum conductance. Yang and Leskovec [87] show that conductance is a good measure to capture the structure of ground-truth communities. For this reason, in our algorithm (Algorithm 4) we only add a candidate node to the community if its addition results in a smaller conductance.

**Density.** We can measure how tightly knit a $C$ is with density, defined as

$$\texttt{density}(C) = \frac{|E(C)|}{|C| \cdot (|C| - 1)} \tag{3.8}$$

**Structural Similarity.** Inspired by Cosine Similarity, the similarity between two nodes $u$ and $v$ can be computed as [34]:

$$s_w(u, v) = \frac{2 \cdot w(u, v) + \sum_{x \in N(u) \cup N(v)} w(u, x) \cdot w(v, x)}{\sqrt{\sum_{x \in N(u)} w^2(u, x)} \cdot \sqrt{\sum_{x \in N(v)} w^2(v, x)}} \tag{3.9}$$

This equation is used later in Section 3.2.5 to define the topical similarity of two nodes. For an un-weighed graph $G(V, E)$, the similarity function can be simplified from the observation that $w(u, v) = 1$ if $(u, v) \in E$ and 0 otherwise, giving:

$$s(u, v) = \frac{|N(u) \cap N(v)|}{\sqrt{|N(u)| \cdot |N(v)|}} \tag{3.10}$$

**Internal & External Similarity.** We can now use the definition of similarity above to define the internal similarity of a cluster $C$:

$$S_{int}(C) = \sum_{(u,v) \in E(C)} s(u,v) \tag{3.11}$$

and the external similarity as

$$S_{ext}(C) = \sum_{(u,v) \in B(C)} s(u,v) \tag{3.12}$$

where $B(C) = \texttt{bridges}(C, V \setminus C)$.

**Tightness.** Huang et al. [35] use the definitions of similarity defined above to introduce *tightness* of a cluster $C$ as:

$$\texttt{tightness}(C) = \frac{S_{int}(C)}{S_{int}(C) + S_{ext}(C)} \tag{3.13}$$

The tightness of a community $C$ will increase when $C$ has high internal similarity and low external similarity.

**Tightness Gain.** In order to use tightness during seed expansion, Huang et al. [35] show that the tightness gain of adding a node $u$ to $C$ can be defined as:

$$\tau(C, v; \alpha) = \frac{S_{ext}(C)}{S_{int}(C)} - \frac{\alpha \cdot S_{ext}(C, v) - S_{int}(C, v)}{2 \cdot S_{int}(C, v)} \tag{3.14}$$

with

$$S_{int}(C, u) = \sum_{v \in N(u) \cap C} s(u,v) \tag{3.15}$$

$$S_{ext}(C, u) = \sum_{(u,v) \in B(C)} s(u,v) \tag{3.16}$$

where $\alpha$ is a parameter which controls the proportion of external similarity of candidate node $v$, larger values of $\alpha$ yield smaller communities [35]. Using Equation 3.14, nodes can be stored in a priority queue, enabling an algorithm to extract the best candidate in $O(1)$ time. Under the conditions where speed is a priority, one could store nodes in a priority queue based on the following score function [32]:

$$\texttt{score}(C, u) = \frac{1}{\texttt{degree}(u)} \sum_{v \in N(u) \cap C} w(u,v) \tag{3.17}$$

Adapted from Equation 3.11, the score is normalised by the degree of the node, this ensures that the score is not biased to those nodes with high degree. Additionally, this normalises ensures that, provided that $0 \leq w \leq 1$ then $0 \leq \texttt{score} \leq 1$. Note that to take structural similarity of nodes into consideration one could use Equation 3.9 as the weight function $w$ or, for a lower computational complexity, it can be defined by an edge similarity based on pre-computed node attributes. Practically, by storing candidate nodes in a priority queue ordered by `score`, we ensure that we are adding nodes in order of their likelihood to take the growing community in a direction where we have larger normalised internal similarity between nodes.

**Updating Scores.** When a new node $z$ is a added to the community, the scores of the nodes in queue must be updated. Fortunately, the only nodes who's scores are updates are the neighbours of $z$, with their scores updated simply as follows:

$$\texttt{score}'(C, u, z) = \frac{1}{\texttt{deg}(u)} \sum_{v \in N(u) \cap C \cup \{z\}} w(u, v) \tag{3.18}$$

$$= \frac{1}{\texttt{deg}(u)} \sum_{v \in N(u) \cap C} w(u, v) + \frac{1}{\texttt{deg}(u)} w(u, z) \tag{3.19}$$

$$= \texttt{score}(C, u) + \frac{1}{\texttt{deg}(u)} w(u, z) \tag{3.20}$$

where $\texttt{score}(C, u)$ is the old pre-calculated score.

### 3.2.5 Topical Quality Measures

To ensure that our clusters are formed with topically coherent content creators, we incorporate topical similarity into our quality measures. In this section we aim to form a weight function $w$ to define the topical similarity of two content creators. We can then use this function in the equations defined in Section 3.2.3. Those which are of particular use are:

**Weighted Structural Similarity.** The weight function can be used in Equation 3.9 to combine both structural similarity (the neighbourhoods) and topical similarity.

**Topical Density.**

$$\texttt{topical-density}(C) = \frac{\sum_{(u,v) \in E(C)} w(u, v)}{|C| \cdot (|C| - 1)} \tag{3.21}$$

**Topical Conductance.** Inspired by the definition of conductance in Equation 3.7 we define a measure the topical conductance of a cluster $C$. We define it as

$$\texttt{topical-cond}(C) = \frac{\sum_{(u,v) \in B(C)} w(u, v)}{\texttt{vol}_w(C)} \tag{3.22}$$

where $B(C) = \texttt{bridges}(C, V \setminus C)$ and

$$\texttt{vol}_w(C) = \sum_{v \in C} \texttt{deg}_w(v) \tag{3.23}$$

with $\texttt{deg}_w$ as a trivial extension of $\texttt{deg}$ by summing the weights of edges as opposed to the number of edges [32].

**Utilising Text.** To determine the topical similarity between two nodes (content creators), we utilise the text which can be derived from the profile. From previous experiments and reasoning in Section 2.2, we decide not use use captions or hashtags as they are often poor topical indicators and provide too much noise. Instead, we use the text in the biography, as we have found this to be the highest quality topical indicator.

**The Problem of Sparsity.** The only prominent issue with using the biography here is sparsity, where some users have empty biographies, or very little text. This prevents us from calculating the topical similarity between content creators where one has a sparse biography.

**Solution.** In order to account for the issue that some users will have sparse biographies we provide some flexibility in our algorithm to allow nodes with no measures of topical similarity to be added to the community. Specifically, we add nodes to a priority queue of candidates using `score` function defined in Equation 3.17 which uses a topically weighted similarity between nodes. In this scenario nodes with sparse biographies will be placed lower in the priority queue, but still will have the opportunity to be considered to be added to the cluster. When a candidate is dequeued, it is only added to the community if the conductance of the community with the node added is smaller than the conductance previous. With this condition, many high-ranking nodes in the queue may be rejected, leaving sparse nodes to be considered. By combining conductance and topical score ordering in this way we are able to both utilise the connections between content creators (Hypothesis 1) with their topical similarity to ensure dense, topically coherent communities are foremd.

**Vectorising Text.** Here we describe how we use state-of-the-art natural language processing techniques to define high-quality feature vectors from the biographical text which allow us to use the hidden semantics of our domain to compute the similarity between two biographies. To obtain high-quality vectors representing the biographies we build a Doc2Vec model (Section 2.4.4) with 500k pre-processed (Section 2.5) biographies of content creators, which captures the semantics of the text in our specific domain. Additionally, we created a bigram phrases [49] model which was used during the pre-processing step which greatly improved the quality of the text features. The details of how we built this model can be found in Section 2.4. With this model we can compute the similarity of two content creators biographies by first inferring their vector representation using our Doc2Vec model, then computing the *cosine similarity* between these vectors. Our methods of computing topical similarity between two content creators is summarised in Algorithm 2.

**Algorithm 2** Computing Topical Similarity of two Biographies

**Input:** Biography of first content creator $b_1$, biography of second content creator $b_1$, infer vector function `vec` using Doc2Vec model
**Output:** Semantic similarity $0 \leq w \leq 1$ of biographies
$t_1 \leftarrow$ pre-process $b_1$
$t_2 \leftarrow$ pre-process $b_2$
$v_1 \leftarrow \texttt{vec}(t_1)$
$v_2 \leftarrow \texttt{vec}(t_2)$
$w \leftarrow s(v_1, v_2)$ using cosine similarity function

### 3.2.6 Determining Optimal Seeds

**Coverage.** To satisfy Criteria 2 and the **high-coverage** requirement outlined in Section 1.4, seed content creators should have different topical content. To ensure that new seed content creators have different topics to those which are already in the seed set, we compare topical features of a content creators. As discussed previously, most profiles contain noisy, sparse and unreliable text. Consequently, if we use the text on the content creators profile we may end up adding a horse-riding content creators who's biography states "I love ice cream" as a seed, when there already exists a horse-riding seed in the set. Instead, we use the title and description metadata of the website linked to on their profile. Our assumptions here are that (i) the website linked to a content creators profile has similar content to the content creators content and (ii) the title and description metadata of websites are more reliable topical indicators. Note we only consider content creators as seeds if they have a website linked to their profile. In conclusion, we first obtain the vector representation of a content creators website and metadata text using text vectorisation methods described in Section 3.2.5. Using these vectors we ensure that a content creator is only added to the seed set if it's vector is sufficiently dis-similar to the content creators' vectors in the current seed set. Dis-similarity is determined experimentally, but typically the threshold resides between 0 and 0.1. This process ensures that each content creator in the seed set are topically independent.

**Overlap and Uniqueness.** To satisfy Criteria 1 and 3 we take a similar approach to [24] by ensuring seeds do not occur in the neighbourhoods of other seeds. The neighbourhood of a seed can be defined as all nodes that can be reached via $h$ hops.

**Heuristic Quality Functions.** To define quality we use domain relevant heuristics. Intuitively one might be tempted to consider page-rank like algorithms to determine the quality of nodes, however this often surfaces fake accounts or celebrities on Instagram, neither of which are good candidates for seeds. In order to define the quality of a seed content creator we must first introduce the idea of engagement. On Instagram, a content creator may create posts (of images or videos), which other users can engage with; by engage we refer to liking or commenting. We can now define a content creator engagement by the proportion of users which follow them that engage with their content, as

$$E(U) = \frac{\sum_{p \in P(U)} (p_{likes} + p_{comments})}{F_{in}(U)} \tag{3.24}$$

where $P(I)$ represents the set of posts of a user $U$, $p_{likes}, p_{comments}$ are the number of likes and comments a post $p$ has, and $F_{in}(U)$ denotes the number of users that follow $U$. To make Equation 3.24 more concise and reduce the computational complexity, we approximate engagement by the following equation:

$$E(u) = \frac{\texttt{ave-likes}(u)}{F_{in}(u)} \tag{3.25}$$

where $\texttt{ave-likes}(u)$ is the average number of likes over $k$ posts. Equation 3.25 is grounded on the assumptions that $p_{likes} >> p_{comments}$ and the standard deviation over the last $k$ posts is low.

$$Q_D(u) = \frac{\delta_{in}(I)}{F_{in}(u)} \tag{3.26}$$

$$Q_E(u) = E(u) \tag{3.27}$$

$$Q_{\hat{E}}(u) = \frac{Q_E(u)}{F_{out}(u)} \tag{3.28}$$

Equation 3.28 captures that a content creator who has high engagement and doesn't follow many people is likely to be of high quality. This helps remove noise but eliminating "fake" content creators who follow lots of people in attempt to get "follow-backs"[1]. Similarly, engagement is taken into consideration to remove users who gained a large following from following users then un-following them soon after. These users will have a large following but low average likes - hence poor engagement ($Q_E$). We also face the challenge of eliminating those users who "purchase" likes, these are called "fake likes". Those users who purchase fake likes have very high "engagement", as a consequence of the number of likes they acquire whilst staying at a relatively low followers size. To account for this we introduce two measures of quality which takes into consideration their followers:

$$Q_{FB}(u) = \frac{1}{\sum_{v \in \delta_{in}}} \mathbf{1}[v \to u] \tag{3.29}$$

where $\mathbf{1}[cond]$ is the indicator random variable which is 1 if $cond$ is true, 0 otherwise and $v \to u$ is true if $v$ follows $u$. Equation 3.29 measures the proportion of a user's followers follow them back, under the assumption that if a user is of high quality then the users which they follow will likely follow them back.

$$Q_{FS}(u) = \frac{1}{\delta_{in}} \frac{1}{F_{in}(u)} \sum_{v \in \delta_{in}} F_{in}(v) \tag{3.30}$$

In Equation 3.30 we determine the average number of followers a user's following has, e.g. if $u$ is followed by $v$ who has 1000 followers and $w$ who has 2000 followers, then $Q_{FS(u)} = 1500$. Finally, we normalise and combine the metrics above to form a function in Equation 3.31 which indicate the importance of a user $u$. The hyper-parameters $\alpha, \beta, \gamma$ and $\delta$ allow us to control how each individual measure of quality

[1] When a user follows another with the aim to get the other to follow them back - not because they authentically enjoy the others content.

effects the final result.

$$Q(u) = \alpha Q_D(u) + \beta Q_{\hat{E}}(u) + \gamma Q_{FB}(u) + \delta Q_{FS}(u) \tag{3.31}$$

Algorithm 3 demonstrates how we use the quality functions we defined in this section to select seeds from a graph of content creators.

**Algorithm 3** Seed Selection Algorithm

**Input:** Graph $G(V, E)$, number of seeds $k$
**Output:** seed set $\mathcal{S}$
  $\mathcal{S} = \emptyset$
  $\mathcal{N} = \emptyset$
  Sort $V$ by $Q$ (Equation 3.31)
  **while** $|\mathcal{S}| < k$ **do**
    $u \leftarrow pop(V)$
    **if** $u \notin \mathcal{N}$ **then**
      $\mathcal{S} \leftarrow \mathcal{S} \cup \{u\}$
      $\mathcal{N} \leftarrow \mathcal{N} \cup \mathbf{N}(u)$
    **end if**
  **end while**

### 3.2.7 Seed Expansion Algorithm

This section outlines the seed expansion algorithm which utilises the insights in Sections 3.2.4 and 3.2.5. Inspired by [32, 35] we greedily select a node from a candidates generated by the neighbours of the nodes in the community. The selection from the candidates is with the maximum score and only added to the community if they improve the conductance of the community. We use conductance to determine whether a node should be added to the community for the reasons discussed in Section 3.2.4 including the fact that Yang and Leskovec [87] show that conductance is a good measure to capture the structure of ground-truth communities. The priority queue in Algorithm 4 is sorted by `score` (Equation 3.17) where the scores of the candidates are updated following the rule described in Equation 3.20. Note that we store node similarities in a cache once calculated for the first time to prevent re-calculation.

### 3.2.8 Evaluation

#### 3.2.8.1 Discovered Topics

**Inferred Topics.** With the community documents we were able to extend our taxonomy to fine grained topics, for instance *Tourism* → *Icelandic Tourism* in Figure 3.2 and *Lifestyle* → *Men's Lifestyle* → *Gentlemen's Lifestyle* in Figure 3.3 (topics which were not present in pre-defined ones).

**Algorithm 4** Topical Seed Expansion Algorithm

**Input:** Initial seed $s$, maximum size of community $k$
**Output:** Topical community $\mathcal{C}$ based on

```
C = {s}
Initialise priority queue Q = N(s)
while |C| < k and do
    n ← pop(Q)
    if conductance(C ∪ n) < conductance(C) then
        C ← C ∪ {n}
        update scores in Q
        Q ← Q ∪ N(n)
    end if
end while
```

**Observations about Text Quality.** Through visual feature extraction in this chapter we were able to provide some evidence to our hypotheses about the text on Instagram. The text relating to the Icelandic Tourism accounts was of higher quality as most of the content-creators were brands and had a requirement to convey themselves clearly on Instagram. Just as frequently we had communities similar to the one in Figure 3.3, where the text is (i) noisy: lack of semantic clarity, the language differs highly from the standard form, (ii) sparse: empty biographies (those which display None) and (iii) inconsistent/unreliable: the text contains more topically unrelated words to those which are topically related (e.g. "Welcome to the club").

**Ground Truth Content Creators.** In addition to discovering topics, we used these communities to select a list of high quality seed accounts for each topic, which are used as ground truth for evaluation purposes in topic inference and classification algorithms in Section 4.5.

*Most frequent terms*

| | word | frequency |
|---|---|---|
| 0 | iceland | 19 |
| 1 | travel | 8 |
| 2 | share | 5 |
| 3 | get | 5 |
| 4 | featured | 5 |
| 5 | tag | 5 |
| 6 | local | 4 |
| 7 | best | 4 |
| 8 | us | 4 |
| 9 | icelandic | 4 |
| 10 | tours | 3 |
| 11 | :camera: | 3 |
| 12 | nature | 3 |
| 13 | use | 3 |
| 14 | pictures | 3 |
| 15 | ❤ | 3 |
| 16 | country | 3 |
| 17 | photos | 3 |
| 18 | food | 3 |
| 19 | official | 3 |
| 20 | offer | 2 |

*Most frequent bigrams*

| | word | frequency |
|---|---|---|
| 0 | best iceland | 2 |
| 1 | iceland offer | 2 |
| 2 | iceland travel | 2 |
| 3 | iceland :camera: | 2 |
| 4 | chance featured | 2 |
| 5 | beautiful country | 2 |
| 6 | iceland use | 2 |
| 7 | trip advisors | 2 |
| 8 | iceland tag | 2 |
| 9 | iceland photos | 2 |
| 10 | koks characterized | 1 |
| 11 | characterized unique | 1 |
| 12 | unique faroese | 1 |
| 13 | faroese identity | 1 |
| 14 | identity commitment | 1 |
| 15 | commitment sustainable | 1 |
| 16 | sustainable local | 1 |
| 17 | local products | 1 |
| 18 | products faroe | 1 |
| 19 | faroe islands | 1 |
| 20 | islands koksrestaurant | 1 |

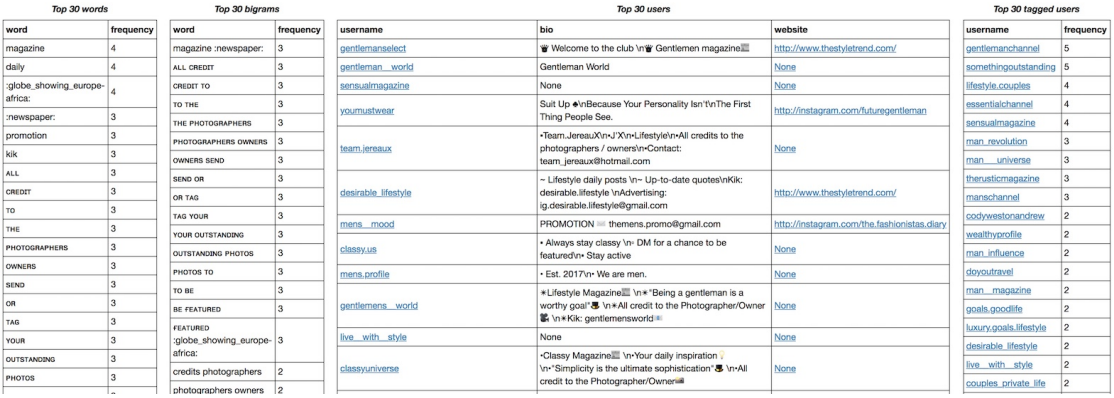
*Top users*

| | username | bio | website |
|---|---|---|---|
| 0 | koks_restaurant | KOKS is characterized by its unique Faroese identity and by its commitment to sustainable and local products. Faroe Islands. #koksrestaurant | http://www.koks.fo/ |
| 1 | icelandnatural | Discover the best of what Iceland has to offer in North America! 🇮🇸 #Iceland #travel \nTwitter: @IcelandNatural\nFacebook.com/IcelandNaturally | http://bit.ly/toi17fall |
| 2 | extremeiceland | 🇮🇸 Extreme Excursions \n📍 Travel Packages - Private Tours\n❄️ Passionate about Iceland \n🇺🇸 Features are welcomed\n🔺 #extremeiceland | http://xtr.is/i |
| 3 | icelandtravel | We're the Iceland Travel experts! See Iceland through our eyes as we capture nature, culture & everyday life. | https://www.icelandtravel.is/blog/detail/nordic-aurora-tips-and-tricks-for-the-perfect-northern-lights-holiday/ |
| 4 | ilovegreenland | Welcome! Use #GreenlandPioneer and #ColourfulNuuk to share your pictures and stories with us and get a chance to be featured! | http://www.greenland.com/ |
| 5 | reykjavikinsider | Are you A blogger coming to Iceland? ! // 🇺🇸 \nGet In Touch Today ❤️ \n👊 #reykjavikinsider | http://www.reykjavikinsider.com/ |
| 6 | asasteinars | 📸 Snapchat: "fromicetospice" \n👧 Ása Steinars\n🌍 Travelblogger from Iceland\n💜 52 country visited\n⬇️ Currently: Iceland | None |
| 7 | icelandadvice | We're an Icelandic travel company.\nTag #icelandadvice for a chance to be featured 👌👊 - Check out our wide selection of tours and activities | https://www.icelandadvice.is/iceland-itinerary/ |
| 8 | iheartreykjavik | Hi there, I'm Auður and I'm mad about my home: Iceland. I want to help you realize how awesome it is too so I created a blog. It's pretty rad. | http://www.iheartreykjavik.net/ |

*Most frequent tagged users*

| | username | frequency |
|---|---|---|
| 0 | icelandair | 5 |
| 1 | guidetoiceland | 5 |
| 2 | wowair | 4 |
| 3 | sorelleamore | 3 |
| 4 | natgeotravel | 3 |
| 5 | wakeupreykjavik | 3 |
| 6 | icelandtravel | 3 |
| 7 | asasteinars | 3 |
| 8 | chrisburkard | 3 |
| 9 | inspiredbyiceland | 2 |
| 10 | wheniniceland | 2 |
| 11 | everydayiceland | 2 |
| 12 | bbc_travel | 2 |
| 13 | earthpix | 2 |
| 14 | visitwestfjords | 2 |
| 15 | igers_iceland | 2 |
| 16 | wisslaren | 2 |
| 17 | horsesoficeland | 2 |
| 18 | mobilemag | 2 |
| 19 | reykjavikfood | 2 |
| 20 | icelandic_explorer | 2 |

Figure 3.2: A community of content creators of topic "Islandic Tourism"



*Top 30 words*

| word | frequency |
|---|---|
| magazine | 4 |
| daily | 4 |
| :globe_showing_europe-africa: | 4 |
| :newspaper: | 3 |
| promotion | 3 |
| kik | 3 |
| ALL | 3 |
| CREDIT | 3 |
| TO | 3 |
| THE | 3 |
| PHOTOGRAPHERS | 3 |
| OWNERS | 3 |
| SEND | 3 |
| OR | 3 |
| TAG | 3 |
| YOUR | 3 |
| OUTSTANDING | 3 |
| PHOTOS | 3 |

*Top 30 bigrams*

| word | frequency |
|---|---|
| magazine :newspaper: | 3 |
| ALL CREDIT | 3 |
| CREDIT TO | 3 |
| TO THE | 3 |
| THE PHOTOGRAPHERS | 3 |
| PHOTOGRAPHERS OWNERS | 3 |
| OWNERS SEND | 3 |
| SEND OR | 3 |
| OR TAG | 3 |
| TAG YOUR | 3 |
| YOUR OUTSTANDING | 3 |
| OUTSTANDING PHOTOS | 3 |
| PHOTOS TO | 3 |
| TO BE | 3 |
| BE FEATURED | 3 |
| FEATURED :globe_showing_europe-africa: | 3 |
| credits photographers | 2 |
| photographers owners | 2 |

*Top 30 users*

| username | bio | website |
|---|---|---|
| gentlemanselect | ♛ Welcome to the club \n♛ Gentlemen magazine | http://www.thestyletrend.com/ |
| gentleman__world | Gentleman World | None |
| sensualmagazine | None | None |
| yourmustwear | Suit Up ♣\nBecause Your Personality Isn't\n\nThe First Thing People See. | http://instagram.com/futuregentleman |
| team.jereaux | •Team.JereauX\n•J'X\n•Lifestyle\n•All credits to the photographers / owners\n•Contact: team_jereaux@hotmail.com | None |
| desirable_lifestyle | ~ Lifestyle daily posts \n~ Up-to-date quotes\nKik: desirable.lifestyle \nAdvertising: ig.desirable.lifestyle@gmail.com | http://www.thestyletrend.com/ |
| mens__mood | PROMOTION themens.promo@gmail.com | http://instagram.com/the.fashionistas.diary |
| classy.us | • Always stay classy \n• DM for a chance to be featured\n• Stay active | None |
| mens.profile | • Est. 2017\n• We are men. | None |
| gentlemens__world | ✶Lifestyle Magazine \n✶"Being a gentleman is a worthy goal" \n✶All credit to the Photographer/Owner \n✶Kik: gentlemensworld | None |
| live__with__style | None | None |
| classyuniverse | •Classy Magazine \n•Your daily inspiration \n•"Simplicity is the ultimate sophistication" \n•All credit to the Photographer/Owner | None |

*Top 30 tagged users*

| username | frequency |
|---|---|
| gentlemanchannel | 5 |
| somethingoutstanding | 5 |
| lifestyle.couples | 4 |
| essentialchannel | 4 |
| sensualmagazine | 4 |
| man_revolution | 3 |
| man__universe | 3 |
| therusticmagazine | 3 |
| manschannel | 3 |
| codywestonandrew | 2 |
| wealthyprofile | 2 |
| man_influence | 2 |
| doyoutravel | 2 |
| man__magazine | 2 |
| goals.goodlife | 2 |
| luxury.goals.lifestyle | 2 |
| desirable_lifestyle | 2 |
| live__with__style | 2 |
| couples_private_life | 2 |

Figure 3.3: A community of content creators of topic "Gentlemen's Lifestyle"

#### 3.2.8.2 Visualisations

Due to the hands on nature of this task, we provide a less rigorous evaluation in this chapter than others. Here we combined a manual exploration approach of evaluation with a numerical one, with emphasis on visualisations to gain an intuition of the results. We used implementations which directly query an embedded Neo4j Graph[2], as opposed to those implemented to read graph data from a file. This was to allow us to skip the (very time consuming) steps of exporting graphs, converting them into a acceptable format then reading the results back into Neo4j. The graph libraries write results back directly to the properties of neo4j nodes, allowing us to export nodes into Gephi format by querying nodes in certain communities with ease. All visualisations were created in Gephi using the Fruchterman Reingold force-directed layout algorithm [23]. It is worth noting that some seeds selected had poor connectedness (from missing data) and so were grown into very small communities.

**Results.** The two visualisations in Figure 3.4 are two graphs of communities generated from two separate seed sets. We make the observation that communities don't tend to join together, which indicates that our methodology of selecting well separated seeds works effectively. Specifically, we selected seeds that don't occur in the neighbourhood of other seeds. Additionally we can see visually that the communities are generally small and tightly knit, which indicates that small, niche communities were discovered.

#### 3.2.8.3 Evaluation Setup

Prior to running the community detection algorithms we determined the largest strongly connected components to remove the possibility of degenerate communities surfacing (e.g. 1-2 node communities). A graph is said to be strongly connected [77] if every node is reachable from every other node. For these experiments we randomly sampled a graph of 500k content creators using the sampling method described below. Our algorithm started with a set of 20 seed content creators for instance, one the seeds is shown in Figure 3.6. These seeds were grown into communities with a specified maximum community size of 300 (parameter $k$ in Algorithm 1.3).

[2] https://github.com/neo4j-contrib/neo4j-graph-algorithms

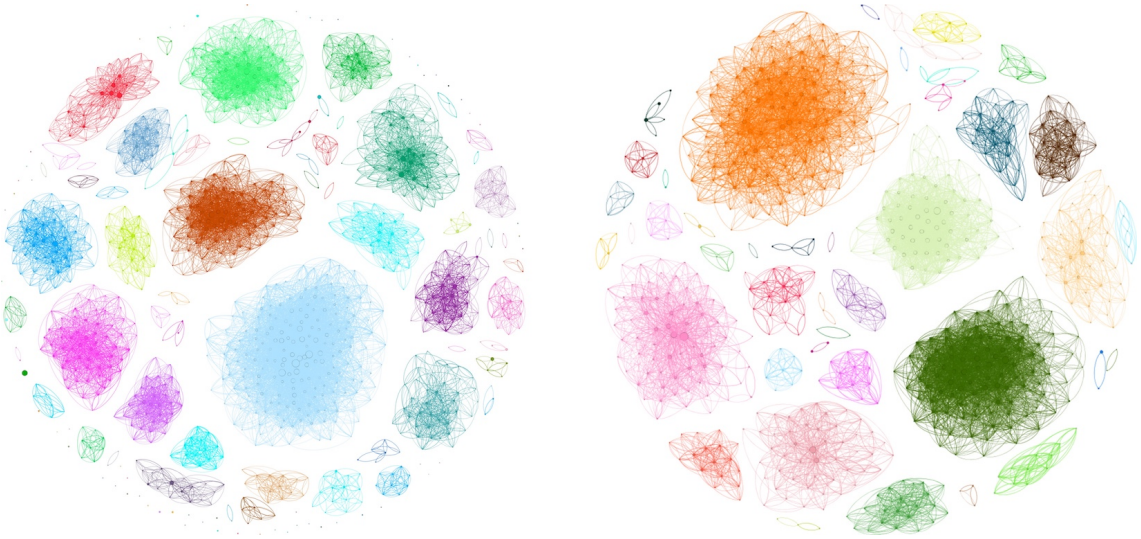


Figure 3.4: In this visualisation of two separate graphs, nodes represent content creators, edges between them represent the "following" relation and a colour represent a community. The two visualisations represent two different sets of communities generated from two different sets of seeds. This visualisation provides evidence that our seed expansion algorithm is able to create well separated communities, i.e. those who are not within each others neighbourhoods, hence providing us the best opportunity to discover niche topics.

**Sampling Method.** We used a similar sampling algorithm to The Forest Fire (FF) [42], inspired by the work on temporal graph evolution. This sampling method randomly picks a seed node and begins "burning" outgoing links. The nodes on the other ends of burnt links have a change to burn their own links. This process is repeated recursively.

**Metrics.** We compare our algorithm to popular community detection algorithms: Label Propagation [65] and Louvain Modularity [6]. To compare the algorithms we use the following metrics:

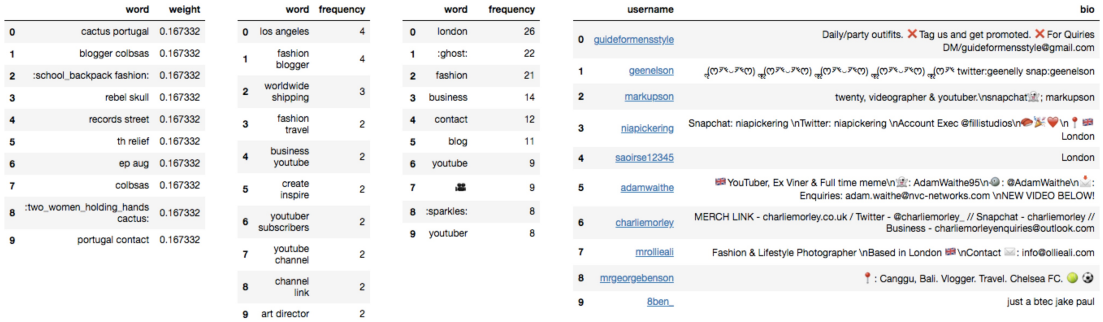
| | |
|---|---|
| *Conductance* | Defined by Equation 3.7, we measure the conductance of the communities as it defines intuitively the quality of a community. Additionally, Yang and Leskovec [87] show that conductance is a good measure to capture the structure of ground-truth communities. |
| *Topical Coherence* | We define topical coherence by Equation 3.32, where $s(u, v)$ is calculated by the cosine similarity of the document embeddings of the biographies of $u$ and $v$ (as described in Algorithm 2). This allows us to measure how well the algorithms generate communities that are able to achieve our goal of topic discovery. |
| *Community Size* | We also measure the average size of the communities generated as it is important that we generate smaller communities for the purpose of niche topic discovery. |

$$coherence(C) = \frac{1}{|C|} \sum_{(u,v) \in E(C)} s(u,v) \tag{3.32}$$

#### 3.2.8.4 Results

**Size and Topical Coherence.** The distribution of size in Table 3.1 show that our algorithm discovers smaller communities on average than the other communities. This is intuitively expected as we set a limit on the size of the communities generated by our algorithm, however, even with the cap on size it still terminates before reaching the maximum community size. Early termination, along with a low topical coherence indicates that it successfully finds communities of niche topical content creators. Contrasting to the other algorithms, the average community size is extremely large, with low average topical coherence. Through these results and visualisations created with Gephi, we conclude that the communities found by traditional CD algorithms are (i) friendship groups and (ii) about general topics. An example of friendship groups can be seen in Figure 3.5, whereby the users are Youtubers who post content about topics ranging from *Fifa* to *Beauty*. Their friendships can be deduced by observing that they co-create Youtube videos together as a part of their "lifestyle" content.

**Conductance.** Our algorithm generates higher conductance than Louvain which is expected as we don't prioritise minimsing conductance in our algorithm. We highlight that we expect that our high coherence can often be caused by adding "popular"[3] users to a community early on in the growth process, thereby drastically increasing the external node degree. When popular users are added, it becomes unlikely that the algorithm will add enough of the neighbours to the community due our sorting on topical coherence which results in a permanently higher conductance.



| | word | weight |
|---|---|---|
| 0 | cactus portugal | 0.167332 |
| 1 | blogger colbsas | 0.167332 |
| 2 | :school_backpack fashion: | 0.167332 |
| 3 | rebel skull | 0.167332 |
| 4 | records street | 0.167332 |
| 5 | th relief | 0.167332 |
| 6 | ep aug | 0.167332 |
| 7 | colbsas | 0.167332 |
| 8 | :two_women_holding_hands cactus: | 0.167332 |
| 9 | portugal contact | 0.167332 |

| | word | frequency |
|---|---|---|
| 0 | los angeles | 4 |
| 1 | fashion blogger | 4 |
| 2 | worldwide shipping | 3 |
| 3 | fashion travel | 2 |
| 4 | business youtube | 2 |
| 5 | create inspire | 2 |
| 6 | youtuber subscribers | 2 |
| 7 | youtube channel | 2 |
| 8 | channel link | 2 |
| 9 | art director | 2 |

| | word | frequency |
|---|---|---|
| 0 | london | 26 |
| 1 | :ghost: | 22 |
| 2 | fashion | 21 |
| 3 | business | 14 |
| 4 | contact | 12 |
| 5 | blog | 11 |
| 6 | youtube | 9 |
| 7 | | 9 |
| 8 | :sparkles: | 8 |
| 9 | youtuber | 8 |

| | username | bio |
|---|---|---|
| 0 | guideformensstyle | Daily/party outfits. ❌Tag us and get promoted. ❌For Quiries DM/guideformensstyle@gmail.com |
| 1 | geenelson | ʕ◔ϖ◔ʔ ʕ◔ϖ◔ʔ ʕ◔ϖ◔ʔ ʕ◔ϖ◔ʔ ʕ◔ϖ◔ʔ twitter:geenelly snap:geenelson |
| 2 | markupson | twenty, videographer & youtuber.\nsnapchat: markupson |
| 3 | niapickering | Snapchat: niapickering \nTwitter: niapickering \nAccount Exec @fillistudios\n\n London |
| 4 | saoirse12345 | London |
| 5 | adamwaithe | YouTuber, Ex Viner & Full time meme\n: AdamWaithe95\n: @AdamWaithe\n: Enquiries: adam.waithe@nvc-networks.com \nNEW VIDEO BELOW! |
| 6 | charliemorley | MERCH LINK - charliemorley.co.uk / Twitter - @charliemorley_ // Snapchat - charliemorley // Business - charliemorleyenquiries@outlook.com |
| 7 | mrollieall | Fashion & Lifestyle Photographer \nBased in London \nContact : info@ollieall.com |
| 8 | mrgeorgebenson | : Canggu, Bali. Vlogger. Travel. Chelsea FC. |
| 9 | 8ben_ | just a btec jake paul |

Figure 3.5: Non-topical community detected by Louvain Modularity CD.

#### 3.2.8.5 Example Community

Here we show an example of a (seed, community) pair, where a "Van Life" content creator @gabriellenelson_ (Figure 3.6) is found during the seed selection process and grown into the community presented in Figure 3.7.

[3]Those with a large degree

Table 3.1: Performance of our LCD algorithm compared to two popular CD algorithms.

| Metric | Algorithm | Mean | Standard Deviation |
|---|---|---|---|
| Topical Coherence | Our Algorithm | **0.6710** | 0.0452 |
| | Louvain | 0.1254 | 0.0311 |
| | LP | 0.2319 | 0.04074 |
| Size | Our Algorithm | **86.5** | 59.54 |
| | Louvain | 7971.76 | 12282.32 |
| | LP | 1973.2 | 3551.4 |
| Conductance | Our Algorithm | 0.6710 | 0.1239 |
| | Louvain | **0.5619** | 0.1064 |
| | LP | 0.7126 | 0.0195 |

## 3.3 Final Taxonomy

**Storage of Taxonomy.** We created the taxonomy in Google Sheets where it can be accessed programmatically via Google Sheets API[7]. This allowed us to create an automatic pipeline which is run whenever there are changes to the taxonomy. This pipeline ensures the changes are propagated through our architecture, the overview of the steps which it performs for a new topic $t$ are outlined below. The details of each step is explained in Chapter 4.

1. Populate concept-topic dictionary with techniques described in Section 4.4.3
2. Generate predicted topics $T_d$ for content creators using concept-topic dictionary classifier
3. Automatically collect labelled websites of topic $t$ and train classifier with new dataset (Section 7)
4. Deploy website classifier
5. Label content creators with that are inferred by ensemble learner

### 3.3.1 Taxonomy Details

This section shows a snapshot of the current taxonomy deployed, it is worth noting that the taxonomy is continuously growing as the team finds new topics and new methods of topic discovery are employed. We attempt to map each topic in our taxonomy to a concept/category on DBPedia, which helps us at later stages define

[7] https://developers.google.com/sheets/api/

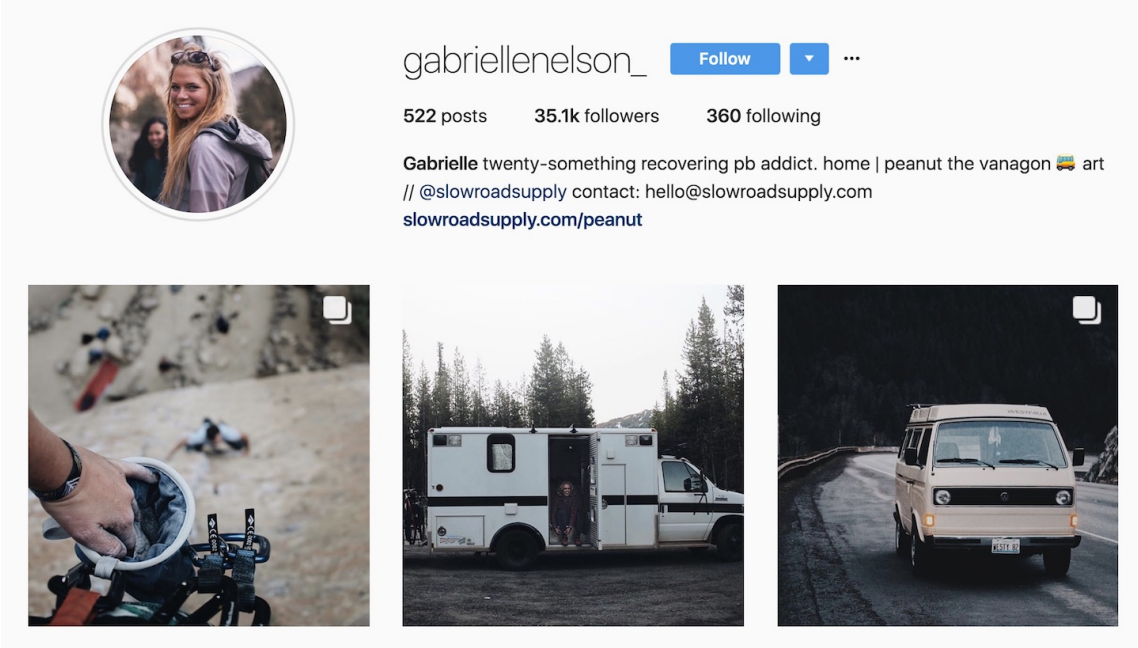


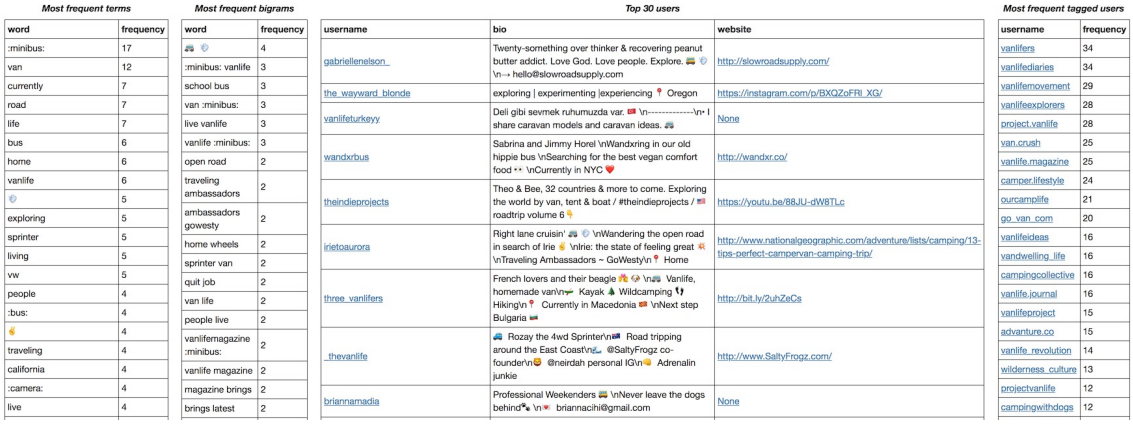

Figure 3.6: A `Van Life` content creator chosen by our seed selection algorithm

semantic similarity between topics by utilising external knowledge bases. An extract of our taxonomy is shown in Figure 3.8. At the writing of this dissertation our taxonomy has the following statistics:

- 431 unique topics
- A maximum depth of 5 tiers, e.g. Arts → Visual Art → Applied Arts → Graphic Design → Typography

## 3.4 Future Work

**Random Walks.** The data used in this work, as discussed previously exhibits a large amount of noise and hence it seems appropriate to expand seeds into communities via random walks. Random walks allows us to capture the small fuzzy topical communities which occur often in Instagram. Here a fuzzy community is one which shows poor community structure. Additionally, Abrahao et al. [1] discovered that communities that are found in in practice are most similar to those that random-walk based algorithms extract. This was achieved by applying a variety of community detection algorithms to a many large scale network datasets. To improve our algorithms performance on detecting ambiguous community structure, inspired by [76, 81], we plan to introduce some randomness into our LCD algorithm to better discover weak communities.



*Most frequent terms*

| word | frequency |
|---|---|
| :minibus: | 17 |
| van | 12 |
| currently | 7 |
| road | 7 |
| life | 7 |
| bus | 6 |
| home | 6 |
| vanlife | 6 |
| 🤍 | 5 |
| exploring | 5 |
| sprinter | 5 |
| living | 5 |
| vw | 5 |
| people | 4 |
| :bus: | 4 |
| ✌️ | 4 |
| traveling | 4 |
| california | 4 |
| :camera: | 4 |
| live | 4 |

*Most frequent bigrams*

| word | frequency |
|---|---|
| 🚐 🤍 | 4 |
| :minibus: vanlife | 3 |
| school bus | 3 |
| van :minibus: | 3 |
| live vanlife | 3 |
| vanlife :minibus: | 3 |
| open road | 2 |
| traveling ambassadors | 2 |
| ambassadors gowesty | 2 |
| home wheels | 2 |
| sprinter van | 2 |
| quit job | 2 |
| van life | 2 |
| people live | 2 |
| vanlifemagazine :minibus: | 2 |
| vanlife magazine | 2 |
| magazine brings | 2 |
| brings latest | 2 |

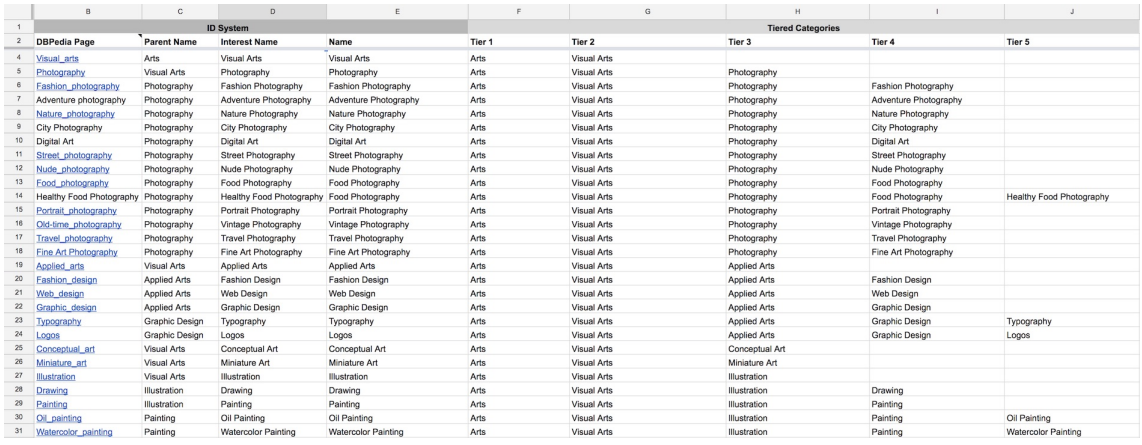
*Top 30 users*

| username | bio | website |
|---|---|---|
| gabriellenelson_ | Twenty-something over thinker & recovering peanut butter addict. Love God. Love people. Explore. 🚐 🤍 \n→ hello@slowroadsupply.com | http://slowroadsupply.com/ |
| the_wayward_blonde | exploring \| experimenting \|experiencing 📍 Oregon | https://instagram.com/p/BXQZoFRl_XG/ |
| vanlifeturkeyy | Deli gibi sevmek ruhumuzda var. 🇹🇷 \n----------\n• I share caravan models and caravan ideas. 🚐 | None |
| wandxrbus | Sabrina and Jimmy Horel \nWandxring in our old hippie bus \nSearching for the best vegan comfort food 👀 \nCurrently in NYC ❤️ | http://wandxr.co/ |
| theindieprojects | Theo & Bee, 32 countries & more to come. Exploring the world by van, tent & boat / #theindieprojects / 🇺🇸 roadtrip volume 6 👇 | https://youtu.be/88JU-dW8TLc |
| irietoaurora | Right lane cruisin' 🚐 🤍 \nWandering the open road in search of Irie ✌️ \nIrie: the state of feeling great 🌴 \nTraveling Ambassadors ~ GoWesty\n📍 Home | http://www.nationalgeographic.com/adventure/lists/camping/13-tips-perfect-campervan-camping-trip/ |
| three_vanlifers | French lovers and their beagle 🐶🐶 \n🚐 Vanlife, homemade van\n🚐 Kayak 🌲 Wildcamping 👣 Hiking\n📍 Currently in Macedonia 🇲🇰 \nNext step Bulgaria 🇧🇬 | http://bit.ly/2uhZeCs |
| _thevanlife | 🚙 Rozay the 4wd Sprinter\n🇦🇺 Road tripping around the East Coast\n🏄 @SaltyFrogz co-founder\n🤤 @neirdah personal IG\n👊 Adrenalin junkie | http://www.SaltyFrogz.com/ |
| briannamadia | Professional Weekenders 🚐 \nNever leave the dogs behind🐾 \n💌 briannacihi@gmail.com | None |

*Most frequent tagged users*

| username | frequency |
|---|---|
| vanlifers | 34 |
| vanlifediaries | 34 |
| vanlifemovement | 29 |
| vanlifeexplorers | 28 |
| project.vanlife | 28 |
| van.crush | 25 |
| vanlife.magazine | 25 |
| camper.lifestyle | 24 |
| ourcamplife | 21 |
| go_van_com | 20 |
| vanlifeideas | 16 |
| vandwelling_life | 16 |
| campingcollective | 16 |
| vanlife.journal | 16 |
| vanlifeproject | 15 |
| advanture.co | 15 |
| vanlife_revolution | 14 |
| wilderness_culture | 13 |
| projectvanlife | 12 |
| campingwithdogs | 12 |

Figure 3.7: A document containing the aggregated features of the `Van Life` discovered community from the seed in Figure 3.6. These aggregated features include the most frequent terms and bigrams[4] in the biographies of the users, the "top users"[5] in the community and the most frequently tagged users[6] by the users in the community



| | ID System | | | Tiered Categories | | | | |
|---|---|---|---|---|---|---|---|---|
| DBPedia Page | Parent Name | Interest Name | Name | Tier 1 | Tier 2 | Tier 3 | Tier 4 | Tier 5 |
| Visual_arts | Arts | Visual Arts | Visual Arts | Arts | Visual Arts | | | |
| Photography | Visual Arts | Photography | Photography | Arts | Visual Arts | Photography | | |
| Fashion_photography | Photography | Fashion Photography | Fashion Photography | Arts | Visual Arts | Photography | Fashion Photography | |
| Adventure photography | Photography | Adventure Photography | Adventure Photography | Arts | Visual Arts | Photography | Adventure Photography | |
| Nature_photography | Photography | Nature Photography | Nature Photography | Arts | Visual Arts | Photography | Nature Photography | |
| City Photography | Photography | City Photography | City Photography | Arts | Visual Arts | Photography | City Photography | |
| Digital Art | Photography | Digital Art | Digital Art | Arts | Visual Arts | Photography | Digital Art | |
| Street_photography | Photography | Street Photography | Street Photography | Arts | Visual Arts | Photography | Street Photography | |
| Nude_photography | Photography | Nude Photography | Nude Photography | Arts | Visual Arts | Photography | Nude Photography | |
| Food_photography | Photography | Food Photography | Food Photography | Arts | Visual Arts | Photography | Food Photography | |
| Healthy Food Photography | Photography | Healthy Food Photography | Food Photography | Arts | Visual Arts | Photography | Food Photography | Healthy Food Photography |
| Portrait_photography | Photography | Portrait Photography | Portrait Photography | Arts | Visual Arts | Photography | Portrait Photography | |
| Old-time_photography | Photography | Vintage Photography | Vintage Photography | Arts | Visual Arts | Photography | Vintage Photography | |
| Travel_photography | Photography | Travel Photography | Travel Photography | Arts | Visual Arts | Photography | Travel Photography | |
| Fine Art Photography | Photography | Fine Art Photography | Fine Art Photography | Arts | Visual Arts | Photography | Fine Art Photography | |
| Applied_arts | Visual Arts | Applied Arts | Applied Arts | Arts | Visual Arts | Applied Arts | | |
| Fashion_design | Applied Arts | Fashion Design | Fashion Design | Arts | Visual Arts | Applied Arts | Fashion Design | |
| Web_design | Applied Arts | Web Design | Web Design | Arts | Visual Arts | Applied Arts | Web Design | |
| Graphic_design | Applied Arts | Graphic Design | Graphic Design | Arts | Visual Arts | Applied Arts | Graphic Design | |
| Typography | Graphic Design | Typography | Typography | Arts | Visual Arts | Applied Arts | Graphic Design | Typography |
| Logos | Graphic Design | Logos | Logos | Arts | Visual Arts | Applied Arts | Graphic Design | Logos |
| Conceptual_art | Visual Arts | Conceptual Art | Conceptual Art | Arts | Visual Arts | Conceptual Art | | |
| Miniature_art | Visual Arts | Miniature Art | Miniature Art | Arts | Visual Arts | Miniature Art | | |
| Illustration | Visual Arts | Illustration | Illustration | Arts | Visual Arts | Illustration | | |
| Drawing | Illustration | Drawing | Drawing | Arts | Visual Arts | Illustration | Drawing | |
| Painting | Illustration | Painting | Painting | Arts | Visual Arts | Illustration | Painting | |
| Oil_painting | Painting | Oil Painting | Oil Painting | Arts | Visual Arts | Illustration | Painting | Oil Painting |
| Watercolor_painting | Painting | Watercolor Painting | Watercolor Painting | Arts | Visual Arts | Illustration | Painting | Watercolor Painting |

Figure 3.8: An extract of our taxonomy stored in Google Sheets.

# High Precision Topic Inference Pipeline

# 4

## 4.1 Introduction

In this section we present a component of our architecture to tackle the problem of acquiring *high-precision* labelled content creators. These content creators are then used in the label spreading algorithm presented in Chapter 5 to spread labels from the small number of labelled content creators across the graph achieving *high-recall*. Consequently, our aim in this section is not to have high-recall, but high-precision. We refer to the labelled content creators acquired in this section as *topical seeds* as they act as seeds in the label spreading stage. This chapter describes an approach to topical classification of social media profiles, combining state of the art approaches from related work and various novel additions which has enabled us to automatically acquire high-precision seeds for each topic in our taxonomy with minimal supervision.

## 4.2 Challenges

### 4.2.1 The Impossibility of Human-Annotation

To ensure for high precision in the label propagation stage of this framework, it is imperative that quality is prioritised over quantity when acquiring labelled content creators. Typically, in this setting one would acquire labelled data through human labelling procedures, for instance crowd-sourced market like Amazon Turk[1]. We reason that this approach is not appropriate to acquire labelled data over a large taxonomy of hundreds of niche topics due to the following reasons:

1. **Expensive.** To obtain accurate labels over a diverse set of topics would require hundreds of hours and millions of pounds.

2. **Cognitive Overload.** As Yang et al. [88] pointed out, large taxonomies present huge cognitive overload where a human annotator would have to retain all topics in memory to identify the relevant ones for each content creator. This would likely result with content creators only labelled with a small set of correct topics.

3. **Human Bias.** It would often be the case that humans would be inclined to label content creators with topics that they are most familiar with, and

[1] https://www.mturk.com

consequently are best at recognising. Additionally, due to the diversity of the taxonomy, many annotators would not be familiar with many topics, hence resulting in a limited set of correct labels.

In conclusion, we decided to not follow the industry standard of human annotation in order to obtain labelled data.

### 4.2.2 Unavailability of Labelled Data

Unlike other OSNs (e.g. Youtube, Pinterest), there is no underlying reliable topic structure, which makes it near impossible to obtain high-quality labelled data directly from the platform. It is worth noting that Instagram provides a means for users to label themselves with a "business category". Despite the existence of business categories, Instagram does not provide features whereby users can find businesses/other users by category. We believe Instagram does not provide this feature due to the issues described in the next section (Section 4.2.2.1). For reasons described in this section we are unable to utilise business categories for high-precision topical inference. Additionally, due to the under-researched nature of Instagram there exists no public pre-labelled datasets, as opposed to Twitter where there are a plethora of methods and datasets to acquire labelled data (Section 2.2.2).

#### 4.2.2.1 Business Categories

Instagram allows business users to assign a category to their account, which indicates to users the category of their "business". Any user can become a business account and assign themselves a category, consequently we found that the majority of content creators and famous content creators have a business category. Unfortunately we discovered through sampling and analysing 500k users with their business categories that there are two highly disabling issues which prevent us from utilising them in our work. These are:

- **Unreliability.** We found that users often inaccurately label themselves. For instance, we found users frequently labelled themselves as topics which they were unrelated to, e.g. a skateboarder labels themselves as "Tourism Company". We manually counted the number of correctly labelled categories in a random sample of 500 accounts and found only approximately 63% were had correct labels.

- **Lack of Specificity.** The popular topics in the taxonomy of business categories lack the depth and specificity we require when labelling content creators. The majority of content creators in a sample of 150k users self-labelled themselves as categories like *Public Figure*, *Community Account* and *Public Blog* (Figure 4.1).

**Conclusions.** Due to the issues described above we were unable to use Instagram business categories for the goal of high-precision niche topic classification. To this end, we devise methods of collecting labelled data through other means.

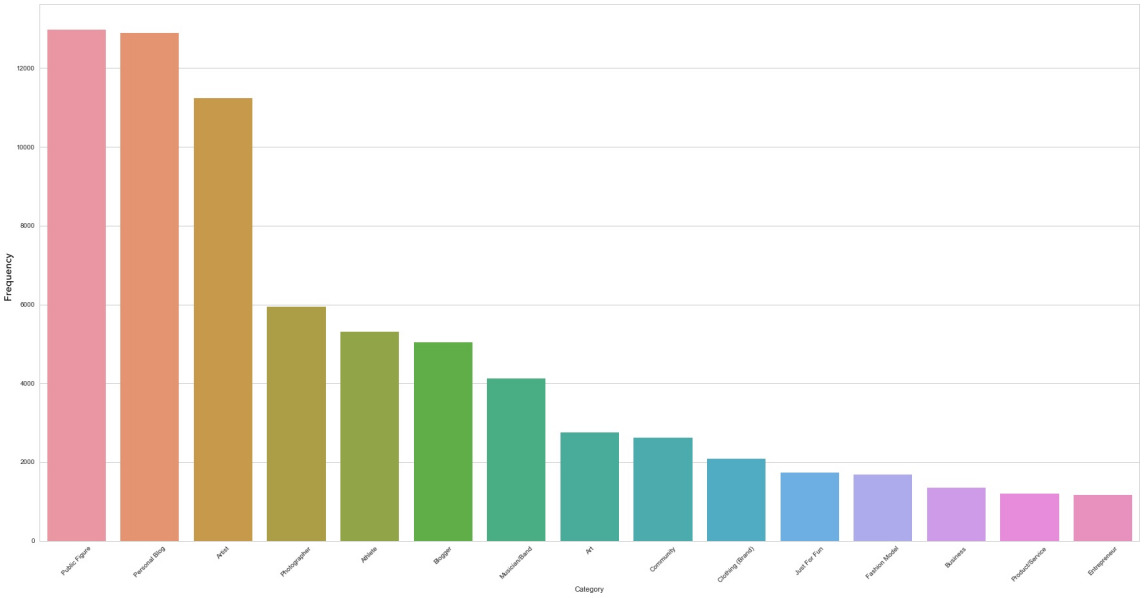



Figure 4.1: Most popular business categories used by a sample of 150k users.

### 4.2.3 Text on Online Social Networks

The issues of text noise, sparsity and unreliability are most prominent in this section, particularly because we're attempting to extract topics from the text on content creators profiles. In this section we briefly give an example of the three challenges associated with text on Instagram and an overview of how we tackle them.

**Noise.** Some common attributes of "noisy" social media text are: spelling mistakes, abbreviations, poor grammar, unicode characters and over-use of punctuation. These issues make it very difficult to generate textual features of the same quality which are used in other text classification domains e.g. news articles or blogs. In an attempt to mitigate the problems associated with noisy text, we first undergo heavy text-preprocessing (including spell checking). Secondly we only consider concepts extracted from the text, limiting the textual features extracted to only the "important" parts of text, thus filtering out a lot of the noise.

**Sparsity.** As Pal et al. [54] noted, Instagram is a visual platform, so algorithms that depend purely on text data do not perform well. The text we are dealing with here is short and often missing. To reduce the effects of sparsity and brevity, we use aggregation techniques, to achieve this we extract the text from the website linked to the profile. For instance if a Youtube account provided as the website, we concatenate the text on the Instagram profile with the text extracted from Youtube profile.

**Unreliability.** This is a unique challenge that is faced by this work, which the task of *user interest inference* (Section 2.2) does not suffer with to the same extent. To demonstrate this, consider a *Poet* who has a biography which includes "Wine lover.". We can confidently infer that the poet's interests include *Wine*, but in the context of topical content classification, *Wine* would be incorrect. Instead we aim to classify the content of the creator as *Poetry*. Pal et al. [54] referred to this issue as *misleading*

*topical signals*. Pal et al. [54] solve this issue by aggregating all the biographies of the users who follow an account then look at the frequencies of topical mentions to assign topics to the user in question. Considering that the average number of follower in our dataset is $64,140$, with just under three million content creators we would have to collect, store and process approximately $192,000,000,000$ users, which make this method of followers aggregation infeasible with our resources. Due to the scale of data required, we take a different approach to solving the issue of *unreliability* in Section 4.4.2.1 which doesn't require the same amount of data and is based on stronger hypotheses with the aim of improving on Pal et al. [54]'s ideas. In addition to aggregation based on *new hypotheses*, we use a hard voting ensemble approach to improve the reliability of our predictions, where we only label a content creator as a topic if multiple classifiers agree on the same topic. These classifiers are built using independent features of a content creators profile, hence provide independent topical signals. We combine these classifiers into a *High-Precision Topic Inference Pipeline* described in Section 4.3.

### 4.2.4 Low Accuracy Concept Extraction

As a part of this section we explore methods similar to those described in Section 2.2.1.1: extracting *primary interests* (topics) from biographies. In these approches, one would attempt to extract keywords/candidates from text and link them to a knowledge base, e.g. extract the bigram "Fashion Photography" and link it to the Wikipedia/DBPedia category `Fashion_photography`. These linked concepts are then used as topics for a user. We describe challenges which directly effect this work, having an impact on high-precision topic extraction. The most common case is a concept is extracted from text which is not relevant to the text, which is caused by poor performance in the entity extraction and linking framework. Incorrect extraction and linking would lead to misclassifying a content creators topic, thereby lowering precision. The most prominent challenges we face are described below:

Table 4.1: Examples of extracted concepts from content creators biographies using TagMe Api.

| Bio | Extracted Concepts |
|---|---|
| Ceramics Paintings Tasmania, Australia Visit website online store | `Ceramic_art`, `Cave_painting`, `Tasmania`, `Australia`, `Online_shopping` |
| Ceramic PRO | `Pottery`, `Azerbaijanis` |
| London based, email to visit studio. I am teaching ceramic workshops in August. | `London`, `Email`, `Ceramic`, `Workshop` |

*In-Accurate* Often concepts will be linked to keywords in the text which are not indicative of that concept. This frequently occurs due to

spelling mistakes on noisy social media text. We dampen the effects of this challenge though using multiple linkers and using the extracted concepts which are most consistent across all linkers.

*False Positives* Using techniques of user interest identification, the last biography in Figure 4.1 would be labelled as the topic. *Computers* from the extracted concept `Email`. In this section we aim to eliminate these false positives through using multiple features from a profile to come to a consensus on "correct" topical labels.

*Large Vocabulary* Multiple concepts relate to the same topic, e.g. in Figure 4.1 three biographies with the same reference to the keyword *ceramic* produced different concepts. We observe this same effect caused by the large and unique vocabulary on Instagram (e.g. slang). The result of this issue comes when attempting to define a mapping between concepts and topics, one cannot simply use the concept alone as the topic, i.e. in the example above, if `Ceramic_art` were used as the topic, then only one of the examples in Figure 4.1 would be labelled correctly. To solve this we undergo manual and automatic methods of building a dictionary, creating a mapping between concepts and topics. This dictionary will contain mappings like `Ceramic_art` → `Ceramics` and `Pottery` → `Ceramics`. The details of the automatic and manual methods we undertook to build this dictionary is described in Section 4.4.

## 4.3 High-Precision Topic Inference Pipeline

### 4.3.1 Topical Signals Identification

A *topical signal* from a content creator is an indication that they post content of certain topics. The indication can be deduced from the information on their profile, for instance the content creators in Figure 4.1 provide topical signals indicating they produce content about *Ceramics*. In order to build classifiers on users features we begin by providing an overview of the topical signals of an Instagram account and means of measuring how accurately they provide an indication of the users topic of content. We acknowledge the existence of other features than those explored in this section, such as images, captions, tagged users and hashtags, but choose to not include them at this stage due to time limitations. It is worth noting, that inspiring work by Pal et al. [54] discovered that the followers of an authority (analogous to content creator in some respects) are good indicators of the topic of the content creator, for instance a footballer will have a large proportion of their audience being interested in football. We explore the topical signal of followers separately in Section 4.4.2.1 whereby we provide several extensions to their approach, forming more reliable signals through stronger hypothesise. Table 4.2 shows the dimensions of an account that we believe are feasible to explore, both computationally and under our time constraints. In this figure we demonstrate topical (`T`) and non-topical (`NT`) examples of each signal, providing the topics which would be inferred all cases.

Table 4.2: Topical Signals that we utilise of an Instagram profile. In this table we contrast between topical and non-topical examples of a signal. A topical signal starts with `T` and non-topical `NT`, for instance "`T` Biography" is a Topical Biography example. The non-topic signals are examples of why a single classifier may perform poorly. The website topics inferred are from the description and title metadata of the website.

| Type | Signal | Example | Topics Inferred |
|---|---|---|---|
| T | Biograpy | Kinderbuch & Editorial Illustration children's books & editorial illustration Harz Mountains, Germany | `Illustration`, `Editorial Illustration`, `Children's Books`, `Mountains` |
| | Website | https://www.juliachristians.de/ | `Illustration` |
| NT | Biography | "Confusingly loud introvert who loves coffee" | `Coffee` |
| | Website | http://www.lizlizo.com/ | Fashion Blog |

**Observations.** The most important observation from this short analysis of topical signals is that there are consistencies amongst the topical signals provided by website and biography of the same user. The examples under the topical signals come from the same user i.e. the first is an illustrator for a children's book. In the typical examples, both the biography and the website's metadata indicate that the most precise user's topical signal is `Illustration`. Additionally, the extracted topic `Editorial Illustration` can be further inferred with higher confidence as it is a sub-category of the extracted topic `Illustration` in the taxonomy. This multi-stage propagation of topics is left to further work. We can further deduce from these examples that there are cases where the topical signals from multiple features of a profile do not align. In these cases we should abstain from classifying the content creator to ensure that our goal of high-precision is met.

**Utilisation of Topical Signals.** Here we provide an overview of how these topical signals can be utilised to infer the topics of a content creator.

*Concepts in Biography.* Concepts are extracted from a biography and classified into topics using a high-precision dictionary approach. For instance the concept `Pottery` in a biography will be classified by the dictionary as the topic `Ceramics`.

*Website.* Instagram users have the ability of entering a url which is displayed on their profile page as their "website". For instance, an account which posts about hand-made cards would use a url to their Etsy[2] shop.

[2] https://www.etsy.com/

Here, we hypothesise that *a content creators website is a good indicator of the content creators topic*. This hypothesis is explored in Section 4.6. In order to determine the topic of the website, we hand-selected a set of common websites which have API endpoints. The API endpoints allows us to deterministically and confidently classify some websites into topics in our taxonomy. An example could include a "Lo-Fi House" music artist who has their website url as an artist page on Spotify, through which we can determine their music genre to be "Lo-Fi House". Websites outside of our hand-picked API set are classified using a state of the art text-classification approach.

*Emojis in Biography.* It is often the case that the use of topical emojis in biographies are a good indicator of the topic of a content creator. This works particularly well with sports, where a basketball player includes basketball relevant emojis in their biography. To infer topics from emojis in our pipeline, we utilised services like EmojiNet [83] to create a whitelisted mapping, converting emojis to potential topics. Due to time restrictions we leave the incorporation of emoji classification to our future work.

### 4.3.2 Pipeline Overview

This section describes an overview of the components of our high-precision topic inference pipeline. Figure 4.2 provides a visual description of how a content creator would be processed by the pipeline. The pipeline comprises of the following components:

**Hard Voting Ensemble.** Identified topical signals often have low-precision. Due to the strict requirement of seeds to have high-precision, we abstain from labelling a content creator if all classifiers don't agree. Specifically, only content creators with a website and a biography are considered. We adopt a hard voting ensemble approach, where each predicted topic $t$ is considered separately and rejected if both Equations 4.1 and 4.2 don't hold. This results in a set of high precision, low recall set of seed content creators for all topics in the taxonomy. The values for $\texttt{min}_w$ and $\texttt{min}_b$ are determined experimentally.

$$p(t|\text{website}) > \texttt{min}_w \tag{4.1}$$

$$p(t|\text{bio}) > \texttt{min}_b \tag{4.2}$$

The equations above indicate the probability of predicting a topic given a website or biography.

**Example Journey of a Classified Content Creator.** For the reader to gain a better understanding of our pipeline we provide an example of how the topic `Illustration` is inferred from a content creator in Figure 4.2 with biography "Kinderbuch & Editorial Illustration children's books & editorial illustration Harz Mountains" and website `https://www.juliachristians.de/`. We encourage the user to follow the example through Figure 4.2. We will refer to the content creator

used in this example as $\mathcal{C}$. The streaming user $\mathcal{C}$ is first checked if they have both a website and non-sparce biography, if this condition is not met, $\mathcal{C}$ is ignored. Otherwise, the website and bio are classified with their corresponding classifiers. With these predictions, we filter out those inferred topics which do not satisfy both Equations 4.1 and 4.2. The remaining topics are then confidently assigned to $\mathcal{C}$ and used to self-train our bio-classifier. An overview of self-training is described in Section 4.3.3.

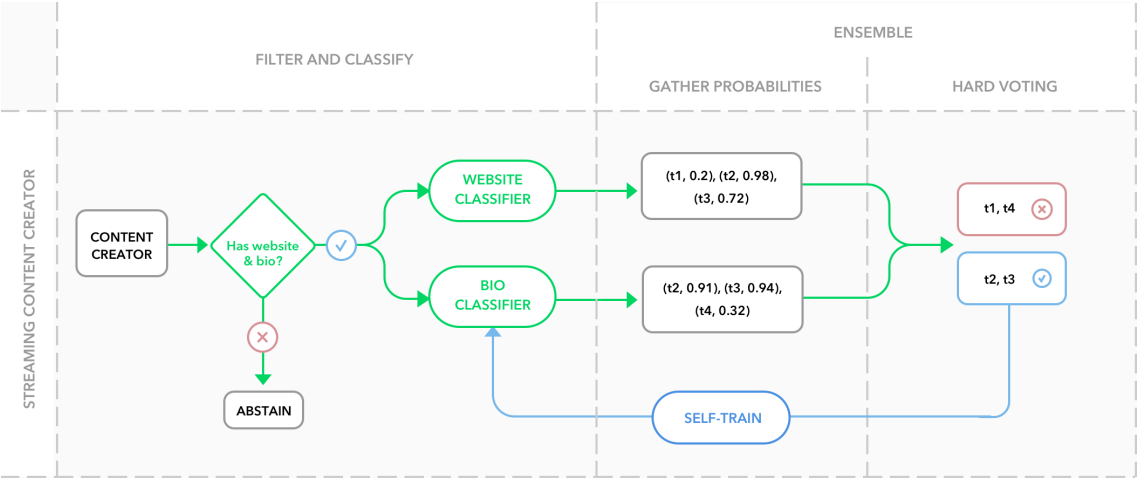



Figure 4.2: Pipeline to achieve high-precision topic inference. A content creator follows through the pipeline left to right, where the topics extracted on the far right in blue are those which the pipeline is sufficiently confident about.

### 4.3.3 Co Training & Self-Training

Previously we described the challenges with acquiring labelled content creators under each category. Labelled content creators are needed to build a biography classifier, which is used in conjunction with other classifiers to achieve the wider goal of high-precision. We utilise two semi-supervised learning methods in order to, with minimal supervision generate labelled content creators to continuously improve the performance of the biography classifier. The two methods we employ are Self Training (using pseudo labels) and Co Training.

**Self Training.** Self-training is the process of using a models output to train itself. By training the model on a small amount of labelled data one can use the initial model to predict the labels of unlabelled data to re-train itself. These predicted labels are called *pseudo-labels*. This process works particularly well when the model predicts with high-precision, which in our case, we take many steps to ensure. An overview of how we employ self training is described in Figure 4.3.

**Co Training.** Co-Training [8] can be used when available data features are redundant and we can train multiple classifiers based on disjoint features. For example, an Instagram profile has a website and biography, which are two disjoint features. The classifiers are then used to train each other. We apply this idea in a similar way to *self training*, whereby the website classifier having access to abundance of labelled data can be used to predict the topic of a content creator, which can be

used as "training data" for the bio classifier.

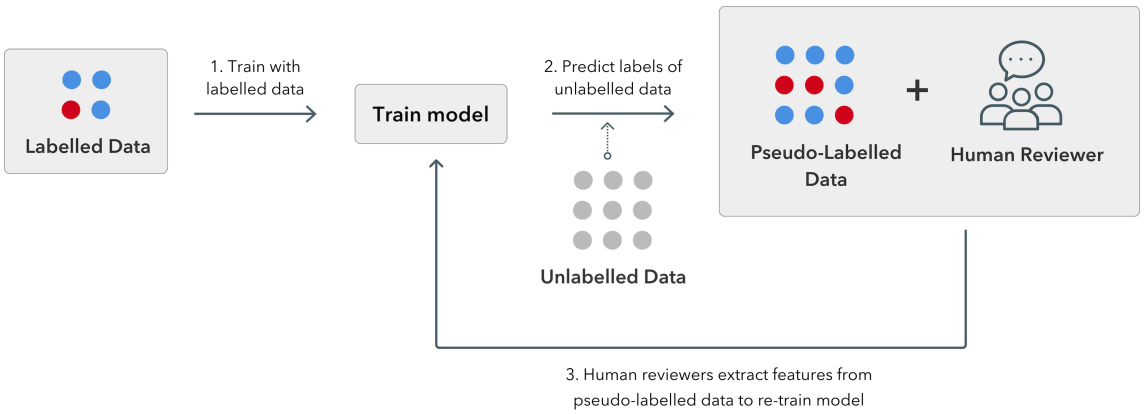



Figure 4.3: A visual demonstration of self training.

**Co-Training in Our Setting.** We identify multiple *independent* features of content creators which can be used as indicators of their topical content. These include biography and website. We utilise this observation, along with the idea of *co-training* to bootstrap the process of classifying the topics biographies in Section 4.4. Specifically we build a website classifier, for which we are able to collect labelled data for and use this to find the predicted topics for all content creators. We then select those which have been classified with sufficiently high confidence to be used as labelled data to "train" our biography classifier. The details of this process is provided in Section 4.4.3.2.

## 4.4 Topical Dictionary Based Classification of Biographies

In this section we describe our methods to generate a dictionary, which creates a mapping from *concepts* which are extracted from biographies to *topics*, e.g. the dictionary might contain an entry mapping the concept *Pottery* to *Ceramics*. This dictionary provides us with a means to extract topical signals from biographies with high confidence.

### 4.4.1 Background

**Linked Open Data and Our Taxonomy.** In our taxonomy we include attributes of each topic. This includes the the DBPedia page/category which relates to each of our topics (equivalent to the Wikipedia page/category name) , e.g. the topic *Nude Photography* will have the attribute *dbpedia* as `Nude_photography`[3].

**Querying Linked Open Data.** This allows us to utilise the power of linked open data to improve our algorithms, for instance knowing that `Nude_photography` is a subject of `Category:Nude_photography` that has broader categories of `Category:Photography` and `Category:Nude_art`. All of the information derived previously can be directly

[3] http://dbpedia.org/page/Nude_photography

```
SELECT ?subject ?label
   WHERE {
   cat:Nude_photography skos:broaderOf ?subject .
   ?subject rdfs:label ?label .
   ?subject rdf:type skos:Concept .
 }
```

Listing 4.1: SPARQL query to get determine the broader subject of the category `Nude_photography`

queried using SPARQL. SPARQL is the most popular query language and protocol for Linked Open Data on the web or for semantic graph databases. It is also commonly refered to as RDF triplestores. Listing 4.1 programmatically demonstrates the example described above using a SPARQL query.

**WordNet and Synonyms.** WordNet [50] is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept [84]. We use the *derivationally related forms* property on WordNet to assist in disambiguating text in biographies. According to WordNet Glossary [79], derivationally related forms are terms in different syntactic categories that have the same root form and are semantically related. We use this feature to normalise terms like "photographer" and "photography" to the same term "photography". We found that applying this normalisation we were able to greatly improve the accuracy of our text models. Additionally we use this database to acquire synonyms for a word which assists us in the process of building our *concept* $\rightarrow$ *topic* dictionary in Section 4.4.3.

### 4.4.2 Extensions to Related Work

#### 4.4.2.1 Authority Discovery

In order to discover the topic of an authority, Pal et al. [54] extract topics from the biographies of all of an authorities followers. These topics are then aggregated together to provide inferred topics with their corresponding probabilities. As discussed previously, the data overhead of this approach is overwhelmingly large. Particularly in our work, where content creators usually have a large number of followers, we would need to collect and processes the followers of many popular accounts. Concretely, the average number of followers in our dataset is $64,140$, with just under three million content creators we would have to collect, store and process approximately $192,000,000,000$ users, which is infeasible with the resources we have. To extent this idea to improve it's effectiveness we present the original hypothesis and present our extension:

**Hypothesis 2** (Pal et al. [54]'s Hypothesis)**.** *An authority on topic $t$ has a significantly higher proportion of followers interested in $t$.*

To present our hypothesis and its expected effectiveness we briefly describe how Pal et al. [54] utilised their hypothesis to perform the sub-task of estimating the topic

of an authority. In their work they used named entity detection to extract entities from biographies which might refer to possible interests and filter the entities using a white listed category set they curated. This whitelisted set is a pruned list of top-level Wikipedia categories. With the extracted topics they compute topic frequencies to assign topics to authorities with probabilities. Other than the scalability problem, we now outline additional issues with this approach and the extensions we propose to mitigate them.

**Issues.** In our work, we aim to determine the topics of content creators of all sizes (5k followers to 20M followers). This presents the possibility of having content creators who have a small number of followers which are predominantly *passive/normal/non-topical* users who's biographies provide a lot of noise. Our introduction discusses in great length that these passive users often exhibit "ideal self-presentation", thereby providing noisy and unreliable topical signals from followers biographies. Through preliminary investigation of a sample of smaller content creators we observed that this approach of aggregation is often inaccurate. An additional observation we make is that there consistently is a proportion of users which follow a smaller content creator who are content creators themselves. Furthermore, these followers who are content creators share very similar topics to the content creator they follow. This follows from the core hypothesis of this work (Hypothesis 1). We summarise this issue as *noisy topical signals from followers.*

**Our Extensions.** To improve the approach we described above we propose an extended hypothesis which provides more robust topical signals and hence *higher precision* in our components.

**Hypothesis 3** (Our Hypothesis - Variation 1)**.** *A content creator of topic $t$ has a significantly higher proportion of followers interested in $t$ who are content creators.*

In summary, we aggregate the topical signals extracted from content creators who follow a content creator in order to infer their topic. We additionally explore the following hypothesis:

**Hypothesis 4** (Our Hypothesis - Variation 2)**.** *A content creator of topic $t$ follows a significantly higher proportion who produce content of topic $t$ content creators.*

Our second variation is one which can be used in the same way our first is. We assess the effectiveness of each hypothesis in the evaluation Section 4.5 by building various models utilising each hypothesis and evaluating with ground-truth data.

### 4.4.3 Dictionary Construction

Here we attempt to address the issues described in Section 4.2.4 where concepts extracted from biographies are often *in-accurate* and have a *large vocabulary.* We tackle these issues with a conservative rule-based dictionary approach similar to Spasojevic et al. [73] described in Section 2.2.1.3. This includes creating a dictionary $\mathcal{D}$ which maps concepts to topics, i.e. $c \rightarrow t$. In order to achieve this we have the challenge of accurately determining which concepts extracted from a biography can

be used as a topical indicators. It is worth noting that this mapping is not known prior to this work, particularly for our custom taxonomy.

#### 4.4.3.1 Automatic Methods using Open Linked Data

As a part of our research we explored the effectiveness of using open linked data (e.g. DBPedia) to increase the recall of our dictionary, effectively solving the *large vocabulary* challenge described in Section 4.2.4. Briefly, this challenge surfaces from the fact that keywords will be linked to concepts in the text which are not a direct match for the topics in our taxonomy (e.g. a mention of the concept `dbpedia:Pottery` refers to the topic `dbpedia:Ceramics`). In order to assist in the generation of our dictionary we explored the following:

*Synonyms.* Multiple concepts relate to the same topic, e.g. in Figure 4.1, three bios with the same reference to the keyword *ceramics* links different concepts. This is as a consequence of the disambiguation processes of the entity linkers we use. We are able to automatically populate our dictionary with synonyms from external databases. In this process we first extract synonyms of a topic in our taxonomy, for instance the synonyms of "ceramics" are {"pottery", "pots", "china", "terracotta"}. Next we link the synonyms to concepts, e.g. "pots" = `dbpedia:Pottery` using techniques described in Section 4.4.4. Finally we create a mapping in our dictionary from the concept its topic, e.g. `dbpedia:Pots` → *Ceramics*. It is worth noting that the concept `dbpedia:Pots` is rarely extracted from biographies, but the concept `dbpedia:Pottery` is, so this mapping has a small number of hits[4]. We provide a solution to this issue with utilising *semantic relationships*.

*Semantic Relationships.* Using relationships in *linked open data* find concepts which may (i) find synonyms and (ii) find the topic (subject) of a concept. The `owl:sameAs` relationship between concepts assist in (i), for example dbpedia:Ceramics `owl:sameAs` dbpedia:Pottery. Additionally the relationships `dbo:wikiPageDisambiguates` and `dbo:wikiPageRedirects` assists in solving the `dbpedia:Pots` issue described above as dbpedia:Pots `dbo:wikiPageDisambiguates` dbpedia:Pottery. Finally the `skos:broader` is useful for (ii).

All mappings extracted from the above methods are added to our dictionary automatically.

**Conclusions.** Using the methods above we are able to generate a dictionary which maps concepts to topics that are within our taxonomy. This dictionary is used as a substitute to poor-precision text classification techniques by utilising linked open data. Using this dictionary, we are able to extract topics from biographies with high-precision. After informal manual evaluation we observed that our dictionary was missing some common concepts which were extracted from the text specific to

[4]Number of times it is used

our domain (Instagram biographies). To solve this issue we devised an algorithm to enhance the dictionary, and improving the frequency it matched concepts extracted from biographies.

#### 4.4.3.2 Bootstrapping and Enhancing the Dictionary with Self-Training and Co-training

We propose Algorithm 5 to enhance our concept-topic dictionary with minimal false-positives. It's important at this point that we highlight that this approach requires some human guidance. The reason we were willing to spend human labour is to ensure that we do not add entries to the dictionary which result in false-positives. Again, due to the requirements of downstream applications for users and businesses, we aim to limit the amount of false-positives as much as possible. Our procedure of enhancing our dictionary can be broken down into the following steps:

**1. Co-Training and Self-training.** Utilising the ideas in Section 4.3.3 we employ a co-training and self-training approach to acquire high-precision labelled content creators from a small amount of initial labelled data inspired by Yang et al. [88]. To employ *co-training* we use our website classification pipeline in Section 4.6 to use one of the *independent features* of an Instagram profile (website). With these classified content creators we select those which are labelled with the highest confidence. We refer to these content creators as *pseudo labels*. To further gather pseudo labels we use *self-training*, whereby the dictionary classifier boostraps itself by continuously attempting to classify content creators, and adding those labelled content creators to the pool of pseudo labels. This process is intuitively named self training as the model gathers labelled data to "train" itself. Concepts are then extracted from high-confidence labelled content creators' biographies. Prior to extracting concepts we pre-process the biographies which involves: removal of links and emails and bigram extraction as described in Section 2.5.0.1.

**2. Concept Aggregation.** We now aggregate extracted concepts from all biographies of content creators in each topic, providing the most frequent concepts for each topic. To exemplify this, imagine a set of content creators were classified using the co-training and self-training described above, with concepts $\mathcal{C}$ extracted from the biographies and grouped by frequency i.e. $\mathcal{C} = \{(c_1, f_1), (c_2, f_2), ...\}$ with pairs corresponding to (concept, freq).

**3. Concept to Topic Dictionary Construction.** Next we hand-extract concepts from $\mathcal{C}$ which are indicative of the topic. Using these hand-picked concepts $C_t = \{c_1, c2, ..., c_k\}$ we create a mapping from $c_k \in C_t \rightarrow t$.

**Example.** To demonstrate this process, we took 135 content creators which were labelled as *Ceramics* by co-training and self-training and found the concepts in their biographies. Figure 4.4 shows the most frequent 8 concepts. From this analysis we manually added a mapping $\{\texttt{Ceramic\_art}, \texttt{Pottery}, \texttt{Ceramic}, \texttt{Ceramic\_engineering}\} \rightarrow \texttt{Ceramics}$. The concept `Ceramic_engineering` is an example of the case we discussed above where concepts were extracted due to the uniqueness of the text in biographies. This mapping would not have been found using the automatic techniques of dictionary generation discussed above.

**Algorithm 5** *Concept* $\rightarrow$ *Topic* Dictionary Construction

**Input:** Topics $\mathcal{T}$, minimum confidence $k$ and current dictionary $\mathcal{D}$
**Output:** Enhanced *Concept* $\rightarrow$ *Topic* dictionary $\mathcal{D}$

```
D = {}
for topic in T do
    U_t ← content creators labelled topic t with minimum confidence k by co-training
    and self-training
    B ← biographies of all content creators in U_t
    Pre-process biographies in B
    C ← concepts extracted from B and grouped by frequency
    C ← filter(C) Here human annotators pick those concepts which correctly
    represent topic
    for c in C do
        D[topic] ← D[topic] ∪ {c}
    end for
end for
```

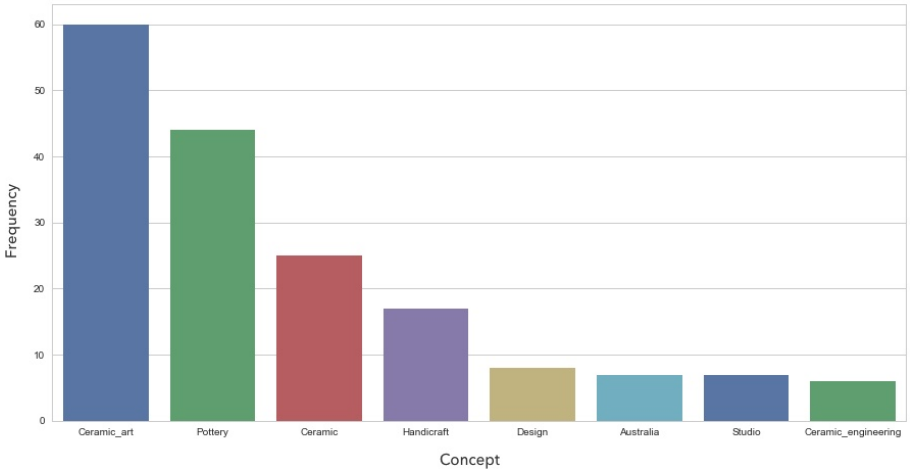



Figure 4.4: Top concepts found in 135 Ceramic content creators biographies.

### 4.4.4 Practical Implementation

At this point we have theoretical methods of generating a dictionary mapping concepts to topics. In this section we describe software engineering techniques to create the components required to practically generate our dictionary.

**Redis.** From [67], "Redis is an open source (BSD licensed), in-memory data structure store, used as a database, cache and message broker". Redis uses key-value stores to store data with pairs of a key and an associated value. This makes it particularly useful to hold our dictionary. Notably, our dictionary will be queried in high-volume so speed is imperative, this makes Redis a much better fit than a NOSQL document database like MongoDB.

**WordNet.** To query WordNet, and subsequently get the *derivationally related forms* and *synonyms* we implemented an API in Python. This API uses NLTK [4]

to query WordNet and Flask[5] to serve the endpoints.

**Concept Extraction.** As described in Section 2.3 we combine multiple entity linkers to improve the results of our entity extraction and linking process. Note that entities are extracted and linked to *concepts*. Specifically we create a Node.js API which acts as a facade, wrapping all the entity linkers into a single API endpoint which we use. Exploiting Node.js' non-blocking I/O model we are able to efficiently gather results from all entity linkers asynchronously, making this approach feasible when requiring high load.

**Text Pre-Processing and Feature Extraction.** To develop text classifiers in this section we employ a pre-processing and feature extraction pipeline similar the one used in our website classification section. For the pre-processing step we first convert tokens to lower case and run them through a stemmer[6]. Following the normalisation steps we remove stop words, punctuation and tokens less than two characters in length. With the pre-processed tokens we build a Doc2Vec model which is used to create feature vectors from the text documents.

**Final Dictionary Classifier.** Our dictionary classifier is deployed to a Node.js backed Express server which exposes an endpoint which takes a user as an input and returns the predicted topics. This application handles collecting aggregated concepts (as described in Section 4.4.2.1) from an efficient Cypher query Listing 4.2. In order to improve efficiency we pre-extract concepts from all content creators biographies and add them as a property to the corresponding node in Neo4j. Once we have obtained the concepts from all neighbourhood biographies of a user the application queries Redis (dictionary) to extract matches to topics. These inferred topics are grouped based on frequency, i.e. $\mathcal{T} = \{(t_1, f_1), ..., (t_k, f_k)\}$ where $t_i$ is an inferred topic and $f_i$ is the frequency for which $t_i$ was inferred from biographies of the users neighbourhood.

**Determining Topics from Predictions.** For evaluation purposes, in Section 4.5 we only attempt to predict one topic for a content creator, under the assumption that they have one predominant topic of content. Hence in our single-prediction models for evaluation purposes we take the first leaf-node predicted topic with the highest probability. In our deployed model and as a part of future work we explore (and will explore) methods of enabling our dictionary classifier to predict multiple topics. To achieve this we began experimenting with two methods of inferring topics from predictions: (i) taking highest weighted probability and (ii) "topic expansion". In this work we provided implementations of these methods but more engineering and research must be undertaken prior to rigorous evaluation. We present the high-level overview of the approaches here:

**(i) Inferring via Probabilities.** The probability/confidence of a topic $t_i$ is calculated as Equation 4.3, as the frequency it was extracted divided by the total number of topics inferred. An important observation is that the biographies of content creators usually include common topics like *fashion* and *lifestyle*, which may lead our

[5] http://flask.pocoo.org/
[6] http://www.nltk.org/howto/stem.html

classifier assigning high probabilities to popular topics for most content creators. To account for this we discount topics by multiplying the result by a scaling factor $\delta_i$. This scaling factor indicates the popularity of topic $t_i$, thereby allowing more niche topics to be assigned. With these probabilities we return the topics with sufficiently high probability ($\geq \epsilon$, determined experimentally).

$$p(t_i|\mathcal{T}) = \delta_i \frac{f_i}{\sum_{(t_j,f_j)\in\mathcal{T}} f_j} \tag{4.3}$$

**(ii) Topic Expansion.** Here we aim to use the predicted topic with the highest confidence to determine whether those topics with lower confidence should returned. We present Algorithm 6 which summarises our implementation of this approach. This approach essentially adds a topic $t_i$ to the final predicted set if $t_i$ has sufficiently high probability in the original predictions and satisfies one of the following conditions: (I) it is sufficiently similar to the topic with highest probability $t_0$ or (II) $t_i$ is beneath $t_0$ in the taxonomy tree. Additionally, inspired by Yang et al. [88] we use *ancestor inclusion*: if content creator has topic $t$, then it should also have topic $t'$ if $t'$ is an ancestor of $t$ in our taxonomy. For example if `Nature Photography` is predicted then `Photography` and `Nature` will be included in the final set of topics. Note that here we refer to an *ontology* as opposed to a *taxonomy*. We aim to completely convert our taxonomy into an ontology in future work as topics often have two parents, similar to `Nature Photography`. For the purposes of research and exploration we created a smaller ontology with some topics in our taxonomy to implement the ideas discussed here. To calculate similarity between two topics we considered two methods:

*Concept Similarity*
: Topics in our taxonomy have associated DBPedia pages, for instance the *Ceramics* has metadata associating it to the DBPedia page `dbpedia:Ceramics`. This enables us to compute semantic similarity between these topics (concepts) by utilising information from external knowledge bases.

*Shared Neighbours*
: Looking at topic similarity from a different perspective we used a social graph based approach to calculate similarity. We achieved this by determining the overlap of content creators in each topic. Specifically we first created a set of content creators which represent each topic which we refer to as $\mathcal{N}_t$. This set was created by expanding each labelled content creator of topic $t$ (in the set $\mathcal{C}_t$) into its neighbourhood and aggregated all neighbourhoods together to form $\mathcal{N}_t$, as shown in Equation 4.4. With a set of content creators for each topic we compute the similarity of two topics as defined in Equation 4.5.

$$\mathcal{N}_t = \bigcup_{c\in\mathcal{C}_t} \texttt{neighbours}(c) \tag{4.4}$$

$$\texttt{sim}(t_1, t_2) = \frac{\mathcal{N}_1 \cap \mathcal{N}_2}{\mathcal{N}_1 \cup \mathcal{N}_2} \tag{4.5}$$

**Algorithm 6** Topic Expansion Algorithm

**Input:** Predicted topics $\mathcal{T} = \{t_1, ..., t_k\}$, their corresponding probabilities $\mathcal{P} = \{p_1, ..., p_k\}$, minimum probability $\epsilon_p$ and minimum topical similarity $\epsilon_s$
**Output:** Final inferred topics $\hat{\mathcal{T}} = \{t_1, ..., t_j\}$

$n \leftarrow len(\mathcal{T})$

**if** $n == 0$ **then**
  **return** $\emptyset$
**end if**

$t_0 \leftarrow \mathcal{T}[0]$
**if** $n == 1$ **then**
  **return** $\{t_0\}$
**end if**

$\hat{\mathcal{T}} \leftarrow \{\}$
$i \leftarrow 1$
**while** $p_i > \epsilon_p$ **and** $\texttt{sim}(t_0, t_i) > \epsilon_s$ **and** $i < n$ **do**
  $\hat{\mathcal{T}} \leftarrow \hat{\mathcal{T}} \cup \{t_i\}$
  $i \leftarrow i + 1$
**end while**

$\hat{\mathcal{T}} \leftarrow \hat{\mathcal{T}} \cup \texttt{include-ancestors}(\hat{\mathcal{T}})$

**return** $\hat{\mathcal{T}}$

```
MATCH u - [:FOLLOWS] - v RETURN v
```

Listing 4.2: Efficient Neo4j Cypher query to find all nodes within the node step neighbourhood of node $n$

From our initial experiments we found that using the *Shared Neighbours* approach to be a more expressive and accurate measure of similarity of topics.

## 4.5 Quantitative Evaluation

### 4.5.1 Setup

**Ground Truth Dataset.** In order to evaluate our methods against those proposed in previous work, we curated a ground truth dataset of labelled content creators. Due to the requirement of high-precision we highlight the importance of having a high-quality ground truth dataset to measure the precision of our models. To achieve this, we hand-curated the dataset with experts at Filli Studios [7] where we peer-reviewed each others topical labels for content creators. As this task by nature is highly time consuming, we limited the topics we would gather content creators for to a randomly selected 20 topics. We then found 150 content creators for each of these topics. We believe a dataset of this size will be sufficient to achieve the goal of comparing our model to previous work and measuring how well we met our goal of high-precision.

**Comparison of Models.** In addition to comparing our model to previous work we use this section to explore variations of hypotheses described in Section 4.4.2.1 to determine which performs best.

### 4.5.2 Precision and Recall Metrics

To measure whether we satisfied our of *high-precision* goal for this section, we quantify out results using the following metrics: *precision*, *recall* and $F_1$ *score*. To ensure for high-precision we employed an abstaining classification approach, whereby our classifiers would not predict anything if they weren't sufficiently confident. The metrics *recall* and consequently $F_1$ *score* do not typically account for missing predictions. Recall, by definition should measure the models ability to to find all relevant data points of interest. To do so, we provide a definition of the unmodified equation of *recall* when applied to content creator topic classification:

$$recall_t = \frac{TP_t}{TP_t + FN_t} \tag{4.6}$$

where $TP_t$ represents the *true positives of topic* $t$: those content creators classified as topic $t$ when their correct topic was indeed $t$. $FN_t$ is the number of *false negatives of topic* $t$: the number of content creators which were labelled as topic $t$ by the

[7] http://fillistudios.com

model, but their topic was another. With this definition (as implemented by all standard libraries like Scikit Learn[8]), the occasions where a content creator is not labelled anything are ignored. This largely over-estimates the value of precision. In our work we use the symbol $\perp$ to represent the empty topic. To measure the empty topic in our metrics we count the number of times a content creator of topic $t$ is classified as nothing ($\perp$), this frequency is calculated for each class and represented by $M_t$. For instance if 10 content creators of label $t$ were not classified as anything, then $M_t = 10$. Furthermore, in this evaluation we use Equation 4.7 as the definition of *recall* of a topic $t$.

$$recall_t^m = \frac{TP_t}{TP_t + FN_t + M_t} \tag{4.7}$$

Our measure of *precision* remains the same as the standard definition where it intends to measure the proportion of data points the model classifies as relevant which were actually relevant. Concretely:

$$precision_t = \frac{TP_t}{TP_t + FP_t} \tag{4.8}$$

where $TP_t$ has the same definition as above and $FP_t$ signifies *false positives of* $t$, which in this work represents the number of content creators of topic $t$ which are not labelled by the model as $t$. With *recall* and *precision* we define $F_1$ *score* which combines both into one metric by taking the harmonic mean of precision and recall. It is defined by the following equation:

$$F_1^t = 2 * \frac{precision_t * recall_t^m}{precision_t + recall_t^m} \tag{4.9}$$

note that we include our definition of recall, thereby taking into consideration the unclassified content creators. Finally, for all topics we average the precision, recall, $F_1$ and missed to provide metrics to measure the overall effectiveness of a model:

$$recall^m = \frac{1}{|T|} \sum_{t \in T} recall_t^m \tag{4.10}$$

$$precision = \frac{1}{|T|} \sum_{t \in T} precision_t \tag{4.11}$$

$$F_1 = \frac{1}{|T|} \sum_{t \in T} F_1^t \tag{4.12}$$

$$missed = \frac{1}{|T|} \sum_{t \in T} M_t \tag{4.13}$$

#### 4.5.2.1 Dictionary Approach Comparisons

Here we present variations of our dictionary model for which we will evaluate in order to determine which performs the best. Each model is grounded upon a different hypothesis, therefore, as a result of comparing the models we determine which hypothesis is strongest.

[8] http://scikit-learn.org/stable/index.html

| | |
|---|---|
| ***Baseline*** *Single User Biographical Dictionary Classifier (`BIO-SIN-DICT`)* | This method extracts concepts from a content creators biography and uses the dictionary built in Section 4.4.3 to infer the topics of a user. This model is used as a baseline as it's the most simplest and that which is used by much of related work. |
| *Followers Biographical Dictionary Classifier (`BIO-AGG-DICT-IN`)* | This method extracts topics of a single user in the same way as `BIO-SIN-DICT`) but does so for all content creators that follow the content creator $U$ attempting to be classified. These topics are aggregated to determine the most likely topic for $U$. The implementation details are described in Section 4.4.4. This is built upon Hypothesis 3. |
| *Following Biographical Dictionary Classifier (`BIO-AGG-DICT-OUT`)* | This model follows the same approach as `BIO-AGG-DICT-IN` except we use Hypothesis 4 by inferring topics from the biographies which the content creator follows. |
| *Neighbourhood Biographical Dictionary Classifier (`BIO-BOTH-DICT-OUT`)* | To predict the topics of a content creator $U$, this model uses the $U$'s neighbourhood of content creators, i.e. all content creators which $U$ is followed by and that $U$ follows. |

#### 4.5.2.2 Results

**Baseline.** Table 4.3 shows the results from conducting the experiments described above. Our first observation is that our baseline model `BIO-SIN-DICT` performs significantly worse than our best performing model. The low recall is the result which we expected due to the sparsity of the text in biographies, where the classifier fails to predict a topic for bios which are empty or have a small amount of text. Additionally, the low precision verifies our hypothesis that the text in Instagram biographies is noisy: the classifier isn't consistently able to find concepts (and consequently topics) in biographies.

**Aggregation Methods.** In comparison to methods which employ aggregation, we see that both precision and recall are improved. This is the result we expected as it mitigates the effects of noise and sparsity by providing more information to the classifiers.

**The Goal of High-Precision.** The results for our best model have exceptionally high precision which verify the success of the steps we undertook to ensure high-precision was achieved. Additionally, we observe that recall is consistently mediocre, which is as expected and will be tackled in Chapter 5.

**Insights and Conclusions.** `BIO-AGG-DICT-BOTH` which uses the neighbourhood around a content creator to form inferences appears to have the lowest precision of the three aggregated models but has the highest recall, indicating larger context provides more opportunity to form inferences but leads to lower precision. `BIO-AGG-DICT-OUT` which utilises Hypothesis 4 outperforms `BIO-AGG-DICT-IN`, which reflects Hypothesis 3 which provides evidence of the superiority of Hypothesis 4. We

conclude with this result that the best topical indicator our of those we have studied is *the content creators which a user follows*. This important observation leads to key design decisions in our recommendation system.

Table 4.3: Precision, Recall and F1 score for the dictionary models over a ground-truth labelled dataset. The recall measured here takes into consideration the empty topic $\perp$ which captures when the model predicts no label.

| Model | *Precision* | *Recall* | $F_1$ |
|---|---|---|---|
| `BIO-SIN-DICT` | 0.82 | 0.28 | 0.42 |
| `BIO-AGG-DICT-IN` | 0.89 | 0.42 | 0.57 |
| `BIO-AGG-DICT-BOTH` | 0.86 | 0.59 | 0.70 |
| `BIO-AGG-DICT-OUT` | **0.92** | **0.56** | **0.70** |

### 4.5.3 Model Comparison

From the issues described in Section 4.2.3 relating to the text in OSNs, we expect a traditional text classification approach to be ineffective in this context. To verify this hypothesis we build two text classifiers in attempt to classify the topical content of an Instagram account and compare the results to the best model in the previous section:

***Baseline*** *Single User Biographical Text Classifier (`BIO-SIN-TEXT`).*

A classifier is trained with the text of a content creators biography. The text is passed through a token-level text-preprocessing and feature generation pipeline described in 4.4.4. We use this model as the baseline as it is the simplest and most widely adopted approach.

*Followers Biographical Text Classifier (`BIO-AGG-TEXT-OUT`).*

As suggested by prior research (Section 4.4.2.1), the aggregation of textual features provides the state-of-the-art interest inference. To extend the ideas in previous work, we use the hypothesis that *a connection between content creators is more indicative of* ***homophily*** *than a connection between normal users*. Specifically, to compare this method to ours we aggregate the biographies of all the content creators which the content creator follows into a document. This document is then used in the same way as `BIO-SIN-TEXT` to generate textual features.

#### 4.5.3.1 Results

**Baseline Observations.** The results of these experiments are presented in Table 4.4. We suspect that the bio text classifier baseline performs worse than than

BIO-SIN-DICT baseline for the dictionary classifier as the model was only trained on a small labelled dataset. We highlight this point as the validation for the reasons we decided to build a dictionary classifier for our final model as opposed to a text classifier: *a dictionary approach performs better than text classification approaches in domains lacking in labelled data.*

**Aggregation Observations.** With the aggregated approaches the model has more data to learn how to predict topics, but still performs worse than our dictionary approach. This again provides evidence for the noisiness of social media text and the effectiveness of the measures we undertook to ensure high-precision. We believe with sufficiently large amounts high-quality training data and a carefully selected and fine tuned text classification pipeline one can achieve good precision with biographical topic classification, however as we have highlighted before, this approach cannot be undertaken in this work due to the inability to acquire labelled data.

Table 4.4: Precision, Recall and F1 score for the best performing dictionary model and bio text classifiers over a ground-truth labelled dataset. The recall measured here takes into consideration the empty topic $\perp$ which captures when the model predicts no label.

| Model | *Precision* | *Recall* | $F_1$ |
|---|---|---|---|
| BIO-SIN-TEXT | 0.43 | 0.28 | 0.34 |
| BIO-SIN-DICT | 0.82 | 0.28 | 0.42 |
| BIO-AGG-TEXT-OUT | 0.62 | 0.31 | 0.41 |
| BIO-AGG-DICT-OUT | **0.92** | **0.56** | **0.70** |

## 4.6 Website Classification

In this section we present the development and evaluation of of a multi-label website classifier. Our method combines website classification with those similar to Kinsella et al. [38], where API endpoints are exploited to accurately extract the topical content of the page, for instance querying the Amazon Product API to determine the category of a book linked by an author. We extend their approaches by utilising external API's to a larger extend and provide state-of-the art classification techniques to predict topics over our custom taxonomy. Our website classification pipeline is outlined in Figure 4.5. In this figure we show the usage of a whitelisted map and blacklist of urls. The blacklist is list of popular websites which provide high false-positive rate or those which are not relevant, e.g. links to Paypal. We also create a whitelisted map, containing entries of the form $(host, topic)$, if the url has a host in the whitelisted map, then it's corresponding topic is returned. The whitelisted map is used when a host has a specific topic but the content on it's page provides no additional information, e.g. 21buttons.com refers to a fashion website but provides no information about the type of fashion.

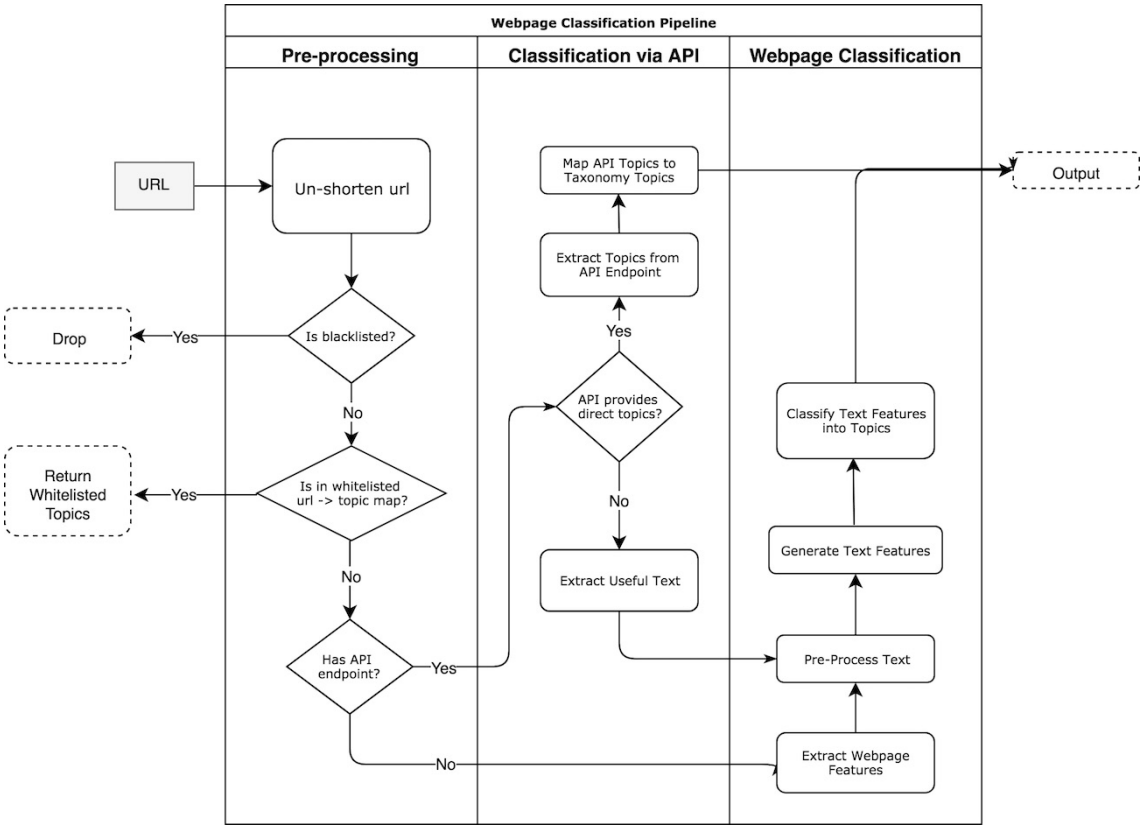



Figure 4.5: Webpage Classification Pipeline

### 4.6.1 Related Work

**User Interest Identification through Hyperlink Extraction.** Kinsella et al. [38] showed that a fruitful attempt to enrich a user interest profile exploits related metadata from hyperlinks to external websites, such as Youtube videos, news articles and Amazon products. Kinsella et al. [38] showed that the use metadata from hyperlinked documents improves topic classification on forums and Twitter. In order to compare the effectiveness of metadata sources they constructed several features from text, for which they would compare the results from classifiers using each feature. Some feature extraction methods discussed: 1. `Content (without URLs)`: original post with hyperlinks removed, 2. `HTML`: text parsed from linked documents and 3. `Metadata`: external meta data retrieved from the hyperlinks of the post. Metadata is extracted using the API of the website linked (if applicable). For example, requesting the Youtube Videos API [9] for the Ted talk video titled *Do schools kill creativity? | Sir Ken Robinson* returns the category `Education` and tags include `TED`, `Talks`, `creativity` and `schooling`. Metadata can be extracted from hyperlinks using domain tagging services. OpenDNS, a crowdsorucing platform provides this service, whereby a community tag domains with category and vote for their correctness, eventually converging to an accurate tag. As an illustration, `nintendo.com` is tagged as *Games* [10].

**Discussion.** Clearly exploiting hyperlinks would improve the performance when

[9] https://developers.google.com/youtube/v3/docs/videos/list
[10] https://domain.opendns.com/nintendo.com

inferring topics of Instagram users, where this method provides the most rich and relevant metatdata when the websites linked provide APIs. This method is less effective on Instagram as it only allows users to post hyperlinks to their biography and/or website[11], as opposed to the captions of their photos. This restricts the amount of hyperlinked metadata that can be extracted from an individuals profile. Fortunately, most users provide hyperlinks in their website field referring to their work/hobbies. This often manifests in the form of their other social media accounts (most often Youtube channels) and personal websites. In our work we take inspiration from Kinsella et al. [38] by utilising API endpoints but do not concatenate the metadata of webpage in the same way they do. Instead we train two separate classifiers: one on the webpage metadata and the other on the text in the profiles biography. As discussed previously, this is to meet our aim of high-precision topic classification via an ensemble classification approach.

### 4.6.2 An analysis of hyperlinks on Instagram

Intuitively, we should expect that the hyperlinks used by content creators should be consistent with their topics of content. Additionally we expect that there are types of hyperlinks which occur more frequently than others, for instance links to Youtube. To explore these hypothesise we randomly sampled 8k content creators who have the website field populated on their profile and performed the following analysis:

- **Hosts Aggregation Analysis.** We mapped the hyperlinks to the host of urls of the form *scheme://host:port/path?query*. For instance the url `https://www.etsy.com/shop/cwpoet?ref=search_shop_redirect` gets mapped to *etsy.com* (with "www." removed). The purpose of this analysis is to determine if there are any common hosts which are used that we can create a high-precision classification components for. An API driven high-precision classification component is a module of the pipeline whereby it takes a url of a specific domain (e.g. Spotify) and attempts to extract topics from the API endpoint associated with the domain. Additionally a component handles determining whether the topic is an accurate signal for the content creator, this process is described below under Section 4.6.3.

- **Per Host Analysis.** Once frequent hosts are determined we manually explore each to determine what information we can extract from the host. For instance we create graphs like those in Figure 4.6, which allow us to decide on which types of links will provide the most *accurate topical signals*. For instance we ignore those Spotify links which are to a users Spotify (Figure 4.6b) as the content creator is likely just sharing their Spotify account, as opposed to providing a signal that they are a music artist. This is an important step in our pipeline to ensure high-precision as these common API links occur very frequently (Figure 4.5).

[11] Users are able to enter a hyperlink which is displayed publicly displayed on their profile as their "website".

Table 4.5: Top 15 hosts which appear in a sample of 8k content creators.

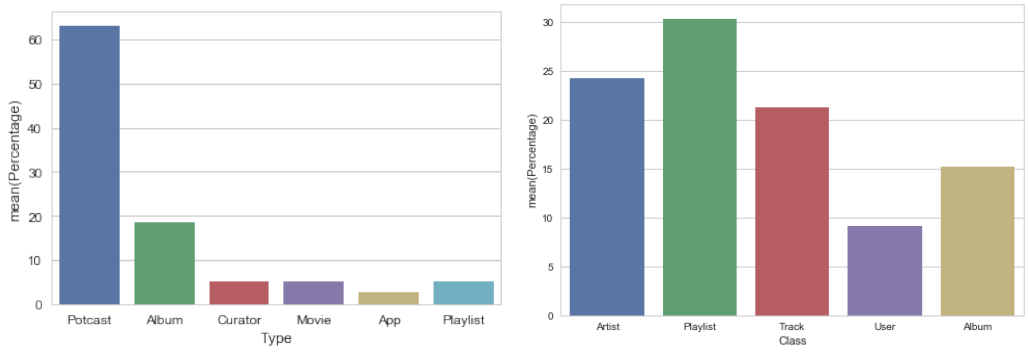
| Host | Frequency |
|---|---|
| youtube.com | 1165 |
| facebook.com | 362 |
| instagram.com | 132 |
| linktr.ee | 130 |
| twitter.com | 92 |
| soundcloud.com | 70 |
| app.21buttons.com | 66 |
| vimeo.com | 58 |
| smarturl.it | 48 |
| amazon.com | 46 |
| itunes.apple.com | 41 |
| open.spotify.com | 37 |
| 21buttons.com | 36 |
| etsy.com | 35 |
| gofundme.com | 31 |

### 4.6.3 Determining Accurate Hyperlink Signals on Instagram

Through exploring the structure of urls we are able to find those which are most likely to have content aligned with the content creators topic. For instance, a content creator may provide a link to Spotify of the structure `https://open.spotify.com/track/XXX` or `https://open.spotify.com/artist/XXX`. We can deduce that the latter is likely to be an artist, whereas the former may be a content creator of another content type posting a track the are fond of. Utilising this observation we are able to query the Spotify API to determine the genre of the artist link, which can be used as the topic of the content creator with high-confidence. In conclusion, we explore various common websites and their url structures in order to generate a set of sub-classifiers for specific websites which we can exploit urls for high-confidence topical labels. Through this process we selected 100 hosts for which we are able to extract accurate topical signals from using a per-host topic extraction module.

#### Insights

- Only 4% of a random sample of 529934 content creators do not have a website linked to their Instagram profile.
- Our of a random sample of 500 content creators, 96% of them have websites




(a) Types of iTunes links.

(b) Types of Spotify links.

Figure 4.6: The most common types of links that are used by content creators within specific hosts. For example, the link type "Podcast" refers to a link with the following form: https://itunes.apple.com/gb/podcast/{podcast-name}/{id}.

relevant to their topic of content. This statistic was arrived at by enumerating the 500 users by hand to determine whether their website is relevant.

- Sparsity of Instagram biographies can be tackled by aggregating information from links to other social media sites like Youtube, Facebook and Twitter. For example Youtube and Twitter both offer biographies which we are able to use in conjunction with the Instagram bio.

### 4.6.4 Automatic Labelled Data Collection

**Demand for Labelled Data.** We use a convolutional neural network to capture the complexity of our data. These complexities lie in our high-dimensional embeddings and in the nuances in differentiating between similar topics. For instance, being able to differentiate between a *Fashion Photographer* and *Fashion Designer*. These models large amount of labelled data in order to capture the complex relations.

**Related Work & Approaches.** Typically one would use publicly available datasets to train a website classifier (e.g. DMOZ [12]). However due to our custom taxonomy, we are faced with a unique challenge of collecting labelled data accross niche categories. Note that in future work we will explore the use of public datasets (e.g. DMOS) to provide labels for some relevant topics.

**Utilising Pinterest.** To tackle this problem we utilise data collected from Pinterest[13]. Pinterest is of particular interest to us as it allows users to search for *Pins* with generic queries like *Handcrafted Jewelry*. A *Pin* is a visual bookmark containing a title, description, a link, and an image or a video [19]. Fortunately, the title and description of a pin are often the metadata from the link the pin is referring to; exactly the features which we use in classifying websites in content creators hyperlinks. This allowed us to collect labelled urls (and their metadata) for each topic in our taxonomy. This process of labelled data collection is summarised in Algorithm 7.

[12] http://dmoz-odp.org
[13] http://pinterest.co.uk/

**Algorithm 7** A module to collect labelled urls (webpages) for each topic in our taxonomy

**Input:** Topics $\mathcal{T}$
**Output:** Labelled urls $\mathcal{U}$
  $\mathcal{U} \leftarrow \{\}$
  **for** *topic* **in** $\mathcal{T}$ **do**
    $\mathcal{P}_t \leftarrow$ all pins from querying Pinterest for *topic*
    **for** $p \in \mathcal{P}_t$ **do**
      $u \leftarrow$ url in $p$
      $t \leftarrow$ title of $p$
      $d \leftarrow$ description of $p$
      $\mathcal{U} \leftarrow \mathcal{U} \cup \{(u, t, d)\}$
    **end for**
  **end for**

### 4.6.5 Webpage Feature Extraction

The textual features of a webpage that we use include the *title* and *description* metadata tags. The reason we decided to only use these is to reduce the complexity of this stage, in order to reach a proof-of-concept. Additionally, this metadata is provided by Pinterest for each pin, so this eliminates the need for us to make a request to each website to gather this information, reducing the computational overhead of this stage. We explore various other methods of feature extraction in future work (Section 4.10.4).

## 4.7 Design Choices

Once the classifier has been built we are able to predict the topic of a website given its description and title metadata. In our website/webpage classification pipeline (Figure 4.5) we first attempt to classify the website by using APIs as they give the most reliable and fine grained topical signals, e.g. an artists Spotfiy page will tell us confidently their music genre but his biography is unlikely to do so. In the cases where the website doesn't fall into our website API classification module we use the text classification model in this section. At this point we make an important observation which we made during our exploration process: *the text from websites provide a significant noise*, this observation is verified in our evaluation (Section 4.8.4). We mitigate this issue in the same way we did for the dictionary classifier, by utilising Hypothesis 4. This leads to our first design choice for this module:

**Design Choice 1.** *To predict the topic of a content creator U utilise the websites of the content creators that U follows.*

**Aggregation Techniques.** Having made the above design choice we must design the best means to use this extra information to improve the *precision* of our predictions. Initially we considered combining the feature vectors extracted from the websites of all the content creators into one feature space. This could be achieved by (i) averaging the document vectors or (ii) simply augmenting the features, i.e. stacking them on top of each other. These methods have some significant drawbacks. Firstly, in (i), by averaging the document vectors we would loose a significant

amount of semantic information (which document vectors were built to preserve!). Secondly, augmenting features in (ii) would result in an inconsistent feature space due to the varying number of neighbours a content creator has, we could consider capping the feature size with padding and trimming techniques, but we would not capture a significant amount of important data. From this reasoning we decide that it is not justifiable to combine feature vectors. To this end, we make our second design choice:

**Design Choice 2.** *Aggregate the noisy predictions from all websites via weighted majority voting. Specifically, we predict the topics of the websites of all content creators which a content creator U follows in a parallel manner. Furthermore, we calculate the probability of U being assigned a topic t based on the frequency of which the topic was predicted. When a users website is classified, we store it in a cache to reduce latency by reducing repeated computations.*

To evaluate the effectiveness of our design choices in this section we compare models that make topic predictions by (i) using only a content creators website and (ii) by using the websites of all content creators the content creators follows. The results from this comparison can be found in Section 4.8.4.

## 4.8 Convolutional Neural Networks for Webpage Classification

The goal of this module of our topic inference pipeline is to (i) automatically collect labelled data for each topic in our taxonomy and use the labelled data to (ii) train a custom designed a website classifier that uses both API endpoints and a one-dimensional convolutional neural network (1D CNN) based on modern NLP pre-processing and feature extraction. It is worth noting that the CNN also acts as a feature-extracting architecture (described below). To provide a detailed report on the process of building and optimising our 1D CNN and feature extraction would be excessive to include in this writeup as it is not the focus of the ideas we aim to convey in this work. To this end, we provide a high-level overview of our techniques and results on building our website classification pipeline. Additionally, we provide details our practical implementation interwoven through this section as opposed to describing the libraries and software engineering in a separate part of this writeup.

### 4.8.1 Convolutional Neural Networks for Text Classification

**CNN's Effectiveness in Text Classification.** Typically the most common and particularly successful text classification approach involves word embeddings to represent words and Convolutional Neural Networks (CNN) for classifying documents. Goldberg [29] describe that the non-linearity of the neural network in addition to the high-quality, easy to integrate pre-trained word embeddings "often leads to superior classification accuracy". Additionally, Goldberg [29] extends his discussion by describing that neural networks with convolutional and pooling layers word particularly well in document classification. This is because they are excellent at extracting salient features (tokens or sequences of tokens) regardless of where they appear in the document. Intuitively, being able to learn what sequences of tokens (words) are

good indicators of a topic, independent of their ordering is incredibly powerful.

**CNNs Introduction.** First we define a convolution in this context as applying sliding window function to a matrix. Now, CNNs can be seen as several layers of convolutions with nonlinear activation functions like ReLU or tanh applied to the results. We refer to a fully-connected layer as a reference to a typical feed-forward neural network layer which connects each input neuron to the output neuron in the next layer. CNNs use convolutions over the input layer as opposed to considering each neuron at a time, this convolution creates the input for the next layer. Pooling[14] is often applied in between the convolution layers to reduce the dimensions of the features. The last layer of a CNN is always a fully connected layer which acts as a classifier with its inputs as the high-level features learnt from the convolutional layers.

**CNNs for NLP Tasks.** Sliding windows are typically applied to a matrix of image pixels for vision tasks. The input to our task (NLP) text classification are sentences or documents represented as a matrix. Each row of the matrix corresponds to one token (usually word), which is represented by the word embedding using a model like word2vec[15] and GloVe[16]. For example, the convolution can slide over windows containing 3 tokens (words) at a time.

### 4.8.2 Tooling, Text-Preprocessing and Feature Extraction

**Pre-Processing.** We begin in a Jupyter Notebook which provided a means for fast data exploration. Using libraries including NLTK[17], Gensim[18] we were able to convert our noisy and diverse text into consistent and clean features. During preprocessing we followed a standard pipeline whereby we removed punctuation, stop words[19] and words with less than 2 characters.

**Feature Extraction and Vectorisation.** It is often recommended that pre-trained word vector models (for instance GloVe) should be used when the amount of data available is small. However due to our specific domain, whereby our vocabulary is sufficiently different to that of the pre-trained models. We suspect that we would have many out of vocabulary words, which are given the same word vector values and hence not capturing all the available information in our documents. Furthermore, limiting the amount of out-of-vocabulary words will greatly improve the accuracy of our models. In conclusion we build our own word vector models using the text collected in Section 4.6.4. We refrain from the details of how we built and evaluated this model as it is not the focus of this work. It is worth noting that research on dealing with out of vocabulary words is an active area [16, 33, 61], which we aim to explore in future work.

[14] http://ufldl.stanford.edu/tutorial/supervised/Pooling/
[15] https://code.google.com/archive/p/word2vec/
[16] https://nlp.stanford.edu/projects/glove/
[17] https://www.nltk.org/
[18] https://radimrehurek.com/gensim/
[19] Stop words are commonly used words, such as "the"

### 4.8.3 Convolutional Neural Network Architecture

In order to utilise the benefits of CNNs for text classification described in this section we use a 1D CNN built in Keras[20] with Theano[21] as the backend. We perform an analysis on our document size and decide on a 5000-word sequence size as the maximum. Those documents which contain more words are truncated, and those with less words are padded with zeros. Given that our word embeddings model has a vocabulary of 100, the input to our CNN should be of dimension $5000 \times 100$. Keras CNN's take an embedding matrix to convert a sequence of tokens (document) into an embedding. We utilise this to provide a means of converting our documents into features which are compatible with CNNs. Algorithm 8 describes our method creating the embedding matrix (similar to that described in an article by Keras [11]).

**Algorithm 8** Embedding matrix calculation to be used to convert text documents into features compatible with a CNN.

**Input:** Word to index map $\mathcal{W}$ with $\mathcal{W}_{len}$ and word embeddings $\mathcal{E}$ of dimension $\mathcal{E}_{dim}$
**Output:** Embedding Matrix $\mathcal{M}$ of dimension $\mathcal{E}_{dim}$ x $\mathcal{W}_{len}$
  Initialise $\mathcal{M}$ with dimension $\mathcal{E}_{dim}$ x $\mathcal{W}_{len}$ as all zeros
  **for** (word, $i$) in $\mathcal{W}$ **do**
    **if** word in vocab of $\mathcal{E}$ **then**
      $\mathcal{M} \leftarrow \mathcal{E}[i]$
    **end if**
  **end for**

### 4.8.4 Evaluation

#### 4.8.4.1 Setup

To evaluate our website classifier we undergo two approaches: (i) evaluate on held-out test set of labelled websites acquired through Pinterest data collection and (ii) hand-curated ground truth dataset of labelled content creators as described in Section 4.5. The purpose of including (ii) as well as (i) is to assess how well the websites on Pinterest generalise to those on Instagram.

**Metrics.** We use the same metrics as the evaluation of our dictionary classifier: *precision*, *recall* (modified to include the empty topic $\perp$) and $F_1$, with a particular focus on precision as it is the requirement of this component to achieve *high precision*.

**Comparisons.** Here we present variations of our website model for which we will evaluate in order to determine the that which performs the best. From the results in Section 4.5, we concluded that the best aggregation model was by using the content creators a user follows. Hence we build an aggregated model by combining the predictions for all the content creators that a content creator $C$ follows in order to infer $C$'s topic.

[20] https://keras.io/
[21] http://www.deeplearning.net/software/theano/

| | |
|---|---|
| ***Baseline*** *Single User Website Classifier (`WEB-SIN`)* | This model uses the content creators website to infer a topic. This model is used as a baseline as it is the most simplest and that which is used by much of related work (only using local features). |
| *Website Classifier on Held Out Data (`WEB-TEST-DATA`)* | This represents the results from evaluating out website classifier on a held-out (30%) dataset of labelled websites collected from Pinterest. |
| *Followers Website Classifier (`WEB-AGG-OUT`)* | This method extracts topics of a single user in the same way as `WEB-SIN`) but does so for all content creators that content creator $U$ follows. These topics are aggregated to determine the most likely topic for $U$. This is built upon Hypothesis 4 and the fact that results in Section 4.5 show that this hypothesis is the best performing. |

#### 4.8.4.2 Results

Table 4.6: Precision, Recall and F1 score for the website classifier models over a dataset of 100 topics collected from Pinterest. The recall measured here takes into consideration the empty topic $\perp$ which captures when the model predicts no label.

| Model | *Precision* | *Recall* | $F_1$ |
|---|---|---|---|
| `WEB-TEST-DATA` | 0.88 | 0.72 | 0.792 |
| `WEB-SIN` | 0.50 | 0.34 | 0.41 |
| `WEB-AGG-OUT` | **0.82** | **0.66** | **0.73** |
| `BIO-AGG-DICT-OUT` | 0.92 | 0.56 | 0.70 |

**Aggregation Model.** Again we observe in Table 4.6 the non-aggregated classifier performs significantly worse than the aggregated one due to noise and sparsity issues due to issues with the title and description metadata of the websites.

**Pinterest data VS Instagram data.** When evaluating our model against our labelled content creators dataset we get similar performance to that of the held-out pinterest data. This indicates that our hypothesis that *hyperlinks in Pinterest pins are similar to those on content creators profiles.*

**Website vs Dict Model.** When comparing our best performing website classification model to the our best performing bio classification model we observe that the bio model performs better. We believe this is due to the following reasons: (i) a simple webpage feature extraction module and (ii) lack of diverse training data. We combat these issues in future work by using more sophisticated feature extraction methods (Section 4.10.4) and other means of labelled data acquisition (Section 4.10.5).

## 4.9 Final Ensemble Model Evaluation

**High-Precision Requirement.** To the end of achieving this components goal of high-precision we combine the classifiers built in previous sections in the form of a *hard voting ensemble*. Concretely, we only classify a content creator as a topic in this component if both the topics predicted using the website classifier and bio dictionary classifier agree.

**Comparison Models.** For comparison models we use the individual classifiers separately. Additionally we create a variation on the ensemble model which predicts the topic that has the highest probability after averaging the probability from each model. We use these comparisons to find the model which produces results with precision as close to 1 as possible.

**Metrics and Dataset.** We use the same metrics for those used when evaluating our website classifier and bio dictionary classifier: *precision*, *recall* and $F_1$ *score*. Similarly, we use the same hand-curated dataset of labelled content creators to evaluate our models on.

### 4.9.1 Results

The results in Table 4.7 shows that ensemble model successfully achieves our goal of high precision, improving on the stand alone classifiers `BIO-AGG-DICT-OUT` and `WEB-AGG-OUT` by at least 0.07 precision. We notice a drop in recall which we expected and sacrifice in order to ensure our the topical seeds provided to our label spreading algorithms are of high quality. From manual exploration we found that this model does not achieve a precision of 1 due to some content creators who are poorly connected and have sparse biographies, i.e. those who follow few other content creators or have content creators in their neighbourhood with sparse biographies. This can be mitigated by abstaining to classify those poorly connected content creators.

Table 4.7: Precision, Recall and F1 score for the ensemble classifier model over a dataset of hand labelled content creators in 20 topics. The recall measured here takes into consideration the empty topic $\perp$ which captures when the model predicts no label.

| Model | *Precision* | *Recall* | $F_1$ |
|---|---|---|---|
| `BIO-SIN-TEXT` | 0.43 | 0.28 | 0.34 |
| `WEB-AGG-OUT` | 0.82 | 0.66 | 0.73 |
| `BIO-AGG-DICT-OUT` | 0.92 | 0.56 | 0.70 |
| `ENSEMBLE` | **0.99** | 0.43 | 0.60 |

## 4.10 Future Work

### 4.10.1 Other Topical Signals

In this component we used three features of an Instagram account to extract topical signals: the biography, website and their social connections to other content creators. To increase both recall and precision we will explore extracting other topical signals from an Instagram account, these include: emojis, captions, images/videos, likes and comments.

### 4.10.2 Multi-Stage Topic Inference

The examples under the topical signals come from the same user i.e. the first is an illustrator for a children's book. In the typical examples, both the biography and the website's metadata indicate that the most precise user's topical signal is `Illustration`. Additionally, the extracted topic `Editorial Illustration` can be further inferred with higher confidence as it is a sub-category of the extracted topic `Illustration` in the taxonomy. This multi-stage propagation of topics is left to further work.

### 4.10.3 Out of Vocabulary Word Embeddings

It is often recommended that pre-trained word vector models (for instance GloVe) should be used when the amount of data available is small. However due to our specific domain, whereby our vocabulary is sufficiently different to that of the pre-trained models. We suspect that we would have many out of vocabulary words, which are given the same word vector values and hence not capturing all the available information in our documents. Furthermore, limiting the amount of out-of-vocabulary words will greatly improve the accuracy of our models. In conclusion we build our own word vector models using the text collected in Section 6. We refrain from the details of how we built and evaluated this model as it is not the focus of this work. It is worth noting that research on dealing with out of vocabulary words is an active area [16, 33, 61], which we aim to explore in future work.

### 4.10.4 Webpage Feature Extraction

With the aim of improving the performance of our website classifier we will explore new methods of webpage feature extraction to provide more complex and rich features to our models. This may include using the content on the page, e.g. the text in important heading tags (`H1`, `H2`). We aim to explore methods like [56, 72].

### 4.10.5 Labelled Website Data Collection

In order to improve the results of of CNN website classifier we aim to continue to devise new methods of collecting labelled data. One such method includes using Google to acquire large quantities of labelled data automatically. We outline the approach we have explored and tend to implement formally:

- Determine popular Google keywords associated with each topic

- Search for the associated keywords in Google and collect the top $n$ results for each topic.
- Download the HTML content for each page and store (HTML, Topic) in database

All (HTML, Topic) pairs can then be processed and used as labelled data for our machine learning models.

# 5 Label Spreading

## 5.1 Introduction

**Objectives.** At this point we have collected a small set of high-quality labelled content creators using our *High-Precision Topic Inference Pipeline*. As shown in the evaluation section, this pipeline achieved high-precision but was unsuccessful in labelling a large enough proportion of the content creators (low recall). This section aims to spread the labels from these initial content creators with graph based semi-supervised learning (GB-SSL) algorithms, utilising strong hypotheses about the structure of our data, thereby achieving high-recall whilst retaining high-precision.

**Semi-Supervised Learning (SSL).** Semi-supervised learning is a class of supervised learning which make use of unlabelled data in training. In a SSL problem, the data $\mathbf{X}$ ($n = \mathbf{X}$) can be divided into two sets of points $\mathbf{X_L} = \{x_1, ..., x_l\}$ and $\mathbf{X_U} = \{x_1, ..., x_u\}$. Where $\mathbf{X_L}$ have a corresponding label set $Y_l = \{y_1, ..., y_l\}$ and the set of labels $\mathbf{Y_U}$ for $\mathbf{X_U}$ are unknown, $y_i \in C$ for a predefined set of classes $C$. Also $n = l+u$ and typically $l \ll u$. The goal of a SSL algorithm is to use the labelled data and relations between labelled and unlabelled data to estimate the labels $\mathbf{Y_U}$.

**Graph Based Semi-Supervised Learning (GB-SSL).** GB-SSL methods are similar to standard SSL methods whereby they utilise unlabelled and labelled data but where the data is represented by a graph, with a node for each labelled and unlabelled example. The graph is usually constructed using standard domain knowledge e.g. $U_1$ follows $U_2$, or defining a similarity metric e.g. $U_1$ and $U_2$ have $N$ likes in common.

**Labelling on Instagram.** GB-SSL algorithms are very suitable to be applied to the propagation of interests on Instagram. This is because (i) it is difficult to collect accurate labels on Instagram, hence only a small proportion of users can be labelled and (ii) the connections between Instagram users (specifically content creators) can be formed into a high quality graph (discussed in Section 5.1.3) describing relationships between labelled and unlabelled users.

### 5.1.1 Assumptions

In comparison to supervised learning (SL) there are certain assumptions which, when held, are utilised by SSL algorithms that use the unlabelled data improve performance on SL algorithms. It might happen that using the unlabelled data

degrades the prediction accuracy by misguiding the inference, which occurs when assumptions are not met. Later we discuss in what ways these assumptions are considered in related literature and how with some reasoning and observations of connections between content creators, they can utilised to improve on past interest analysis literature. We now provide an overview of the assumptions these algorithms are grounded upon and how we can satisfy them maximally in our context.

**The Continuity/Smoothness Assumption.** If two points $x_i$ and $x_j$ are close, then the should share the same label or have similar labels $y_i$ and $y_j$.

**Assumption 5.1.1** (Semi-Supervised Continuity Assumption)**.** *If two points $x_i$ and $x_j$ in a high-density region are close, then they should share the same label or have similar labels $y_i$ and $y_j$.*

**The Cluster Assumption.** If new points which have the same or similar labels form clusters we could use unlabelled data to find the boundary of each cluster more accurately.

**Assumption 5.1.2** (Cluster Assumption)**.** *The data tend to form discrete clusters, and points in the same cluster are more likely to share a label.*

### 5.1.2 Content Creators and Influencers

Up to now, we have defined content creators on Instagram as those users producing high-quality content in a specific topic. From a different perspective, they can be seen intuitively as *social influencers*, whereby they have a (large) audience for whom they are able to persuade by virtue of their authenticity and reach. In this section, we consider *Instagram influencers* and *Instagram content creators* as synonyms.

### 5.1.3 Homophily vs Influence

Here we discuss how the above assumptions apply in the context of Instagram in order to apply GB-SSL algorithms within Instagram. By reviewing how different literature has attempted to satisfy the assumptions and through social science theory we will establish a design decision which will be conducive to highly accurate GB-SSL algorithms. To start we outline prerequisite definitions.

**Homophily** is the property also known as "birds of a feather flock together", which describes that people who share similar attributes group together. These attributes include interests, age, gender and geographical location. This principle has been studied in the context of social networks extensively [45], establishing that similarity breeds connection. Let $U_1$ and $U_2$ be individuals, then Homophily states that:

$$\text{SimilarAttribtes}(U_1, U_2) \rightarrow \text{Connected}(U_1, U_2) \tag{5.1}$$

where $\text{SimilarAttribtes}(U_1, U_2)$ is a binary function which is interpreted as $U_1$ and $U_2$ have similar attributes.

**Social Influence** can occur when one's emotions, opinions, or behaviours are affected by others. Within the context of Instagram, we argue that a user in the large

majority of cases, follow an influencer who they do not know if they are interested in the topic of the influencers content. For example, someone would only follow a graphics designer that they didn't know if they were interested in graphics design. Additionally, influence3rs commonly only follow a select few users. From observations of influencers we have observed that in most cases they follow a small carefully selected number of other users which are highly indicative of their interests and/or content they upload, e.g. graphics design influencers tend to follow other graphics design influencers. We reason this is most likely because they often do not follow friends/family members and use Instagram as a creative platform, hence seek for inspiration related to their creative expertise. More concretely, this phenomenon can be summarised as follows: let $I$ be an social influencer and $U$ be another user, then:

**Assumption 5.1.3.** $\mathit{Influences}(I, U) \rightarrow \mathit{InterestedInContent}(U, I)$

where InterestedInContent$(U_1, U_2)$ is a binary function which reads $U_1$ is interested in the content of $U_2$ and a stronger assumption about influencers:

**Assumption 5.1.4.** $\mathit{Influences}(I, U) \wedge \mathit{isInfluencer}(U) \rightarrow \mathit{SimilarInterests}(I, U)$

where SimilarInterests$(U_1, U_2)$ is a binary function which is interpreted as $U_1$ and $U_2$ have similar interests.

**Application of Homophily to Instagram.** It has is been proposed previously that GB-SSL algorithms like label spreading are used to propagate labels across a graph of connected user nodes through the following relation. As discussed above, these algorithms only perform well when specific assumptions are satisfied. In the context of OSNs, Assumptions 5.1.2 and 5.1.1 have commonly been accepted to hold with a belief that the principle of Homophily is a sufficient justification. However, Homophily provides an implication in the reverse direction to what Assumption 5.1.2 requires. More specifically, the common mis-interpretation [80, 86] of Homphily amongst literature is that:

**Assumption 5.1.5.** $\mathit{Connected}(U_1, U_2) \rightarrow \mathit{SimilarInterests}(U_1, U_2)$

We reason, from intuition and experience that:

**Assumption 5.1.6.** *Within popular OSN, users are less often connecting with people with shared interest and more so with friends and family.*

Under Assumption 5.1.6, people who are connected do not necessarily share similar interests because they are simply using the platform as a social means to communicate with their friends and family. Therefore, both the robustness of Assumption 5.1.5, and consequentially the performance of GB-SSL algorithms which utilise it degrade. From these observations, we form the design choice:

**Design Choice 3.** *For GB-SSL algorithms to produce the best results, assumptions 5.1.2 and 5.1.1 have to hold as strongly as possible. In order to achieve this, we will use only a graph of influencers, where connections are described by the "following" relation. This leads from from assumption 5.1.4, concluding that influencers would mostly be inclined to follow one another if they were interested in each-others content (e.g. golf influencers mostly follow other golf influencers). This produces a high quality graph based on strong assumptions.*

To further validate Design Choice 3, imagine a GB-SSL algorithm is applied to the whole user graph: normal users follow family and friends, which forces the algorithm to propagate interest topic labels amongst friends and family. Clearly this is not often the case, especially in terms of creative interests, hence the algorithm would have propagated labels incorrectly.

**Exploiting Influencers to Determine Interests.** From assumption 5.1.6 and equation 5.1.3, it is possible to conclude that a users interests are defined most accurately by not the users they follow but by the influencers they follow. To re-iterate, in most cases, users follow influencers (content creators) because they are interested in the influencers topical content. Hence this leads to:

**Design Choice 4.** *Once the label spreading has completed on the influencer graph, one can infer the interests of a normal user by grouping the topic labels of the influencer the user follows. This insight is utilised in in our recommendation system, for example U follows influencers who's labels are* `nature photography`*,* `landscape photography`*,* `fitness` *and* `calisthenics`*. we would infer that U's interests are a combination of the labels.*

We now move onto discussing more formally, algorithms to perform label propagation across the graph of influencers.

### 5.1.4 Types of Semi-Supervised Learning

Given a set of $l$ examples $x_1, \ldots, x_l \in X$ with corresponding labels $y_1, \ldots, y_l \in Y$ and $u$ unlabelled examples $x_{l+1}, \ldots, x_{l+u} \in X$:

**Definition 2** (Transductive Learning)**.** The goal of transductive learning is to infer the labels for the given unlabelled data $x_{l+1}, \ldots, x_{l+u}$.

**Definition 3** (Inductive Learning)**.** The goal of inductive learning is to infer the function $f : \mathbb{X} \to \mathbb{Y}$, which can be used to predict the labels of a new $x$.

Transductive learning cannot predict the labels of new data after training, to overcome this, a variety of methods can be used. For instance, a K-Nearest Neighbours approach takes a new $x$, and infers its label as a combination of the labels of its neighbours in the graph. This will be explored further during the implementation stages.

### 5.1.5 Graph Construction

Graphs provide a representation of verticies and relationships between them. Here we will define two types of graphs, **binary** and **weighted**.

**Definition 4** (Binary Graph)**.** Let $G = G(V, E)$ be an undirected graph with edges $E$ and vertices $V$ and adjacency matrix $\mathbf{A} \in \{0, 1\}^{|V| \times |V|}$ where

$$A_{ij} = \begin{cases} 1 & \text{if } (i, j) \in E \\ 0 & otherwise \end{cases}$$

**Definition 5** (Weighted Graph)**.** Let $G = G(V, E)$ be an undirected graph with edges $E$ and vertices $V$ and adjacency matrix $\mathbf{A} \in \mathbb{R}^{|V|\times|V|}$ where.

$$A_{ij} = w(i, j)$$

for some weight function $w$ which defines how similar nodes $i, j$ are. The weights are usually constructed as a symmetric *affinity* matrix $W \in |V|\times|V|$ where $w_{ij} = w(i, j)$.

### 5.1.6 Hard Clamping Label Propagation

Zhu and Ghahramani [92] provide a popular label propagation algorithm where node's labels are propagated to neighbouring nodes according to some distance function $d_{ij}$ defining the distance between nodes $i$ and $j$. It also ensures the original labels are clamped at each iteration, which act as stable sources that propagate labels through unlabelled nodes.

**Preliminary Notation.** Let the labelled data $\{(x_1, y_1), ..., (x_l, y_l)\}$ be in the form (feature, label) for $y \in C$ where $C$ is the set of predefined labels. Also let the unlabelled data (features) be denoted as $X_U = \{x_{l+1}, ..., x_{x+u}\}$, where $l \ll u$ and $n = l + u$. In this setup, similarity between nodes will be represented by the Gaussian kernel function.

**Probabilistic Transition Matrix.** Now the weights between nodes has been established, it is possible to define a probabilistic transition matrix. This matrix $P \in \mathbb{R}^{n\times n}$ which encodes $T_{ij}$ as the probability to jump from node $u$ to $j$:

$$P_{ij} = P(i \to j) = \frac{w_{ij}}{\sum_{k=1}^{n} w_{ik}} \tag{5.2}$$

Let $Y \in [0, 1]^{n\times C}$ where a row is interpreted as the probability distribution over labels for a node. Additionally, we defined $Y_U$ be as a $u \times C$ matrix and $Y_L$ as a $l \times C$ matrix which represent the labels of nodes in the unlabelled and labelled sets respectively.

**The Algorithm** Zhu and Ghahramani [92] describes the algorithm is as follows:

1. Propagate $Y \leftarrow PY$
2. Row-normalise $Y$
3. Clamp the labelled data. Repeat from step 1 until $Y$ converges.

In Algorithm 9, line 5 propagates labels using probability distribution matrix defined in Equation 5.2 and normalises $Y$. This essentially spreads labels along the *local structure*. The normalisation ensures that the rows of $Y$ represent a distribution over classes. Line 6 ensures initial labels are preserved.

**Fixed Point Solution.** The algorithm was proved in the original paper to converge. It was also shown to have a fixed point solution. Let $\bar{P}_{ij} = P_{ij}/\sum_k P_{ik}$, i.e. the row normalised matrix of $P$, $Y_L$ be the $l \times C$ matrix formed by the top $l$ rows of $Y$ (the labelled data) and $Y_U$ be the $uu \times C$ matrix of the remaining $u$ rows. Finally, split $\bar{T}$ after the $l$-th row and $l$-th column into 4 sub matrices

**Algorithm 9** Label Propagation algorithm proposed in [92]

**Input:** Affinity matrix $W$
**Output:** Label Probability Matrix $\hat{Y}$

1: Compute diagonal degree matrix $D$ by $D_{ii} \leftarrow \sum_j W_{ij}$
2: Initialise $\hat{Y}^{(0)} \leftarrow (y_1, ..., y_l, 0, 0, ..., 0)$
3: Compute probabilistic transition matrix $P \leftarrow D^{-1}W$
4: Iterate
    5: $\hat{Y}^{(t+1)} \leftarrow P\hat{Y}^{(t)}$
    6: $\hat{Y}_l^{(t+1)} \leftarrow Y_l^{(0)}$
7: until convergence

$$\bar{T} = \begin{bmatrix} \bar{T}_{ll} & \bar{T}_{lu} \\ \bar{T}_{ul} & \bar{T}_{uu} \end{bmatrix} \tag{5.3}$$

Now from the proof in [92], the fixed point equation to determine the labels $Y_U$ is

$$Y_U = (I - \bar{T}_{uu})^{-1}\bar{T}_{ul}Y_L \tag{5.4}$$

**Discussion** This method penalises any label assignment where two nodes connected by a highly weighted edge are assigned different labels. Consequently, label propagation aims to *smooth* labelling over the graph.

### 5.1.7 Soft Clamping Label Spreading

Zhou et al. [89] proposed a similar model to the basic Label Propagation algorithm in Section 5.1.6, but uses affinity matrix based on the normalised graph Laplacian and soft clamping across the labels. This soft clamping differs from the previous algorithm in a key way: the labels of nodes $X_L$ are allowed to change during label propagation, it doesn't enforce smoothing across the graph. There is a potential for noise in the data from the labelling stages: a small proportion of labels may be incorrect or missing. Hence, by allowing labels to change slightly, may cause the algorithm to converge to a more accurate labelling, this will be explored through experimentation.

**Algorithm 10** Label Spreading Algorithm proposed in [89]

**Input:** Affinity matrix $W$ with $W_{ii} \leftarrow 0$ and parameter $\alpha \in (0, 1)$
**Output:** Label Assignment Matrix $\hat{Y}$

1: Compute diagonal degree matrix $D$ with $D_{ii} \leftarrow \sum_j W_{ij}$
2: Compute normalised graph Laplacian $L \leftarrow D^{-1/2}WD^{-1/2}$
3: Initialise $\hat{Y}^{(0)} \leftarrow (y_1, ..., y_l, 0, 0, ..., 0)$
4: Iterate
    5: $\hat{Y}^{(t+1)} \leftarrow \alpha L\hat{Y}^{(t)} + (1 - \alpha)\hat{Y}^{(0)}$
6: until convergence

### 5.1.8 Regularisation Framework

A graph regularisation method introduced by Zhou et al. [89] based on the soft clamping iteration algorithm introduced in 5.1.7. Regularisation refers to the process of finding a function (in our case matrix $\hat{Y}$) which minimises a set of conditions (objective function). Concretely we can define the cost function as follows,

$$C(Y) = \frac{1}{2}\left( \sum_{i,j=1}^{n} W_{ij} \left\| \frac{1}{\sqrt{D_{ii}}} Y_i - \frac{1}{\sqrt{D_{jj}}} Y_j \right\|^2 + \mu \sum_{i=1}^{n} \left\| Y_i - Y_i^{(0)} \right\|^2 \right) \tag{5.5}$$

where $\mu$ is the regularisation parameter and $Y^{(0)}$ is the initial label assignment matrix. Now the most optimal label assignment matrix is

$$Y^{\star} = \arg\min_{Y} C(Y) \tag{5.6}$$

In Equation 5.5, the left and right hand terms correspond to the *smoothness* and *the fitting* constraints, respectively. The smoothness constraint indicates that points close to each other should have similar labels. The fitting constraint means good classification should not change too much from the initial label assignment. Graph Laplacian regularisation is effective because it constrains the labels to be consistent with the graph structure.

## 5.2 Influencer Graph Structure

From the results in Chapter 4 we saw that the neighbourhood of an influencer acts as a strong topical signal. To explore this further we created the visualisation in Figure 5.2. The nodes in this visualisation represent influencers, where the coloured ones are labelled by topic using the *high-precision topic inference pipeline* built in the previous section. Those which are *gray* are *unlabelled* and were added to the graph by taking the neighbours of the labelled influencers.

### 5.2.1 In Accordance with GB-SSL Assumptions

The visualisation in Figure 5.2 demonstrates evidence that the assumptions required by GB-SSL are satisfied:

- **Continuity Assumption:** *If two points $x_i$ and $x_j$ in a high-density region are close, then they should share the sample label or have similar labels $y_i$ and $y_j$.* It can be observed that labelled influencers appear closer to each other if they have the same label than those who's label differs. Moreover, groups of influencers with the same label appear close to other groups of influencers with similar labels, for instance *Ocean*, *Nature Photography* and *Surfing* all appear near one another.

- **Cluster Assumption:** *The data tend to form discrete clusters, and points in the same cluster are more likely to share a label.* It is clear that the influencers have formed discrete clusters of topics where nodes within the same cluster (in most cases) have the same topic label.

### 5.2.2 Challenges

Additionally, this visualisation provides an introduction into the types of challenges which are faced in this chapter.

**Under Represented Topics.** If a topic is un-represented then an influencer will be labelled the closest to what is captured. For instance, if *Doughnut Baking* is not represented in the taxonomy but *Baking* is, then a *Doughnut Baker* influencer would be labelled as baking from the information the graph has. On exploring the graph in Figure 5.2 and manually exploring the classifications generated by label propagation tests we discovered the majority of incorrect classifications were due to the issue described here. This can be mitigated by iteratively improving the coverage of our taxonomy and propagating new topics through our architecture.

**Similar Topics.** Figure 5.2 shows that both plant based food and baking lie very close to each-other, which makes the algorithm less effective in classifying between them. We deem this acceptable in some scenarios as some bakers labelled by our topic inference pipeline may be plant based bakers. To mitigate this in the scenarios where the topics are not shared we aim to gather a sufficient amount of topical seeds, hypothesising that with more nodes, there will be more opportunity for clearer structure, i.e. independent clusters to form.

**Popular Accounts/Topics.** Popular users[1] within popular topics[2] tend to connect different clusters of topics which provide noise in the label spreading. This is caused by the global nature of the algorithm, where labels will be spread from one topical community through famous users into other topical communities. In this work we remove nodes with high degree from our graph for which we spread labels. The threshold for which we remove nodes is determined by determining the degree distribution of nodes taking the upper $x$ percentile of the distribution. We determine $x$ experimentally depending on the graph, for instance Figure 5.1 shows the degree distribution of the graph in Figure 5.2, where the top 99th and 98th percentiles constitutes to 2.2% and 10% respectively.

**Label Flexibility.** Often influencers will have inferred topics which are accurate from a textual standpoint but are in-accurate from a human subjectivity standpoint. Graph based data provides a different view into the topics of an influencer which may lead to discrepancies amongst components of our architecture. For instance, our topic inference pipeline inferred the topic *Interior Design* @perfectioninmotion's, however *@perfectioninmotion* is placed in a cluster of car content creators in the following graph as *@perfectioninmotion* is a *automotive wall decor* content creator and hence follows a large number of car content creators. If we were to use the *hard clamping* variation of LP then our algorithm will only allow *@perfectioninmotion* to be labelled as *Interior Design*. This leads to the following design choice:

**Design Choice 5.** *Allow initial labelling of seed content creators to change through a soft clamping propagation approach.*

[1] Those users who are followed by a large number of users
[2] Those topics which are enjoyed by a variety of audiences

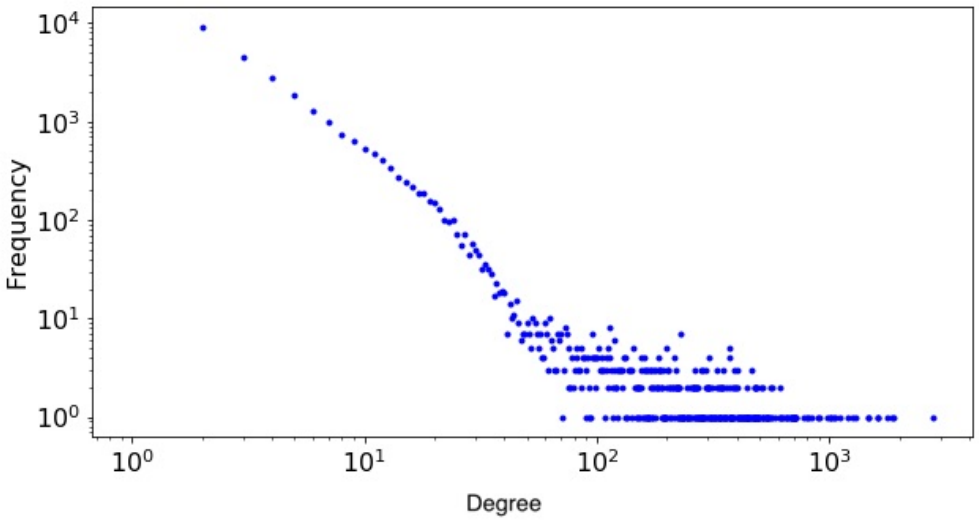



Figure 5.1: The degree distribution of the graph in Figure 5.2
.

*Soft clamping* enables the initial label assignments to change, allowing *@perfection-inmotion* to be labelled as both *Cars* and *Interior Design* based on the initial labels and the labels propagated. This idea is explored in our evaluation (Section 5.3)

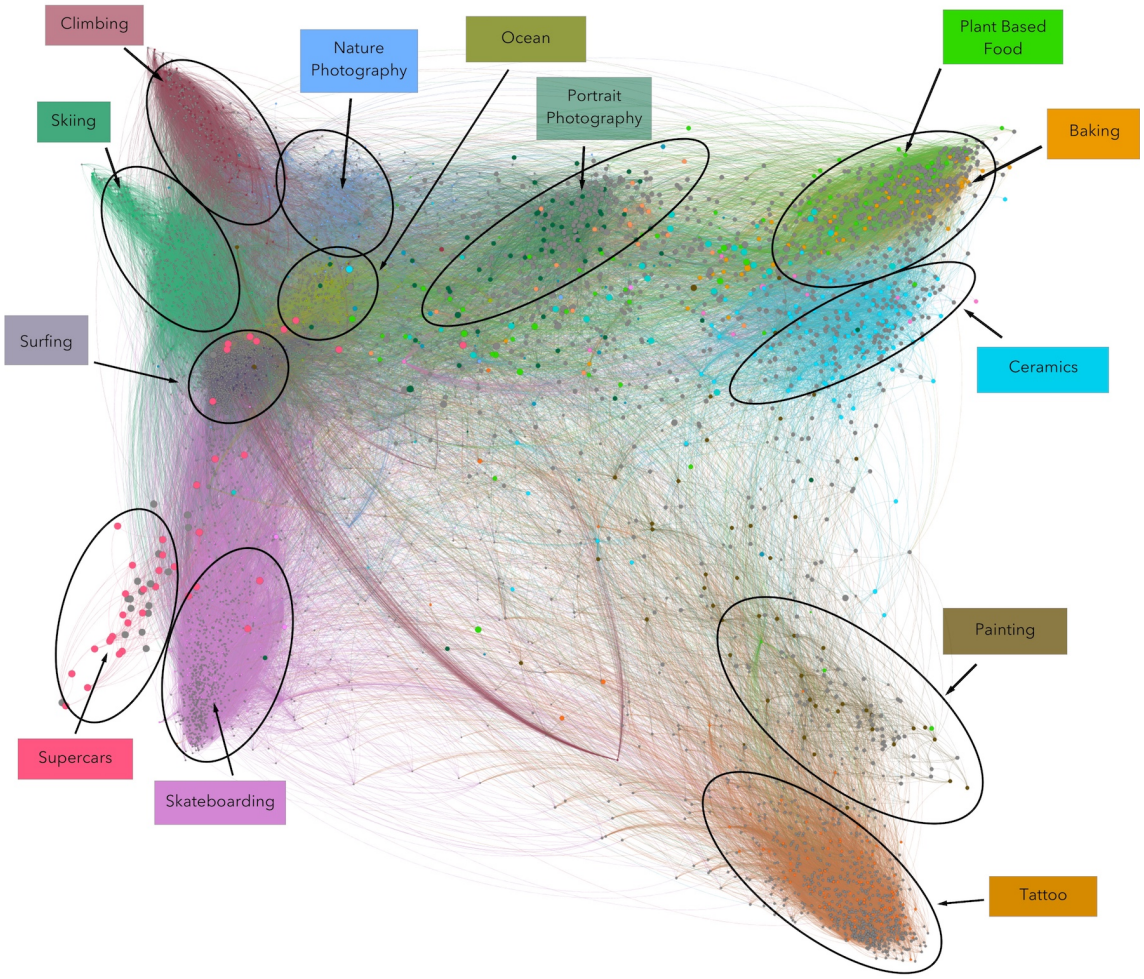



Figure 5.2: A topical community graph of influencers as nodes and the edges between representing the "following" relation, visualised with a Force Layout [23]. The graph was generated by starting with a small number of labelled influencers and adding the influencers in their neighbourhoods. This visualisation shows each topic as a separate colour, with influencers and their edges coloured by their labelled topic. A nodes size is reflective of its degree[3]. The large proportion of the graph is *unlabelled*, with the colour grey.

## 5.3 Evaluation of the Label Spreading Algorithm

In this evaluation we aim to assess the *recall* of the standard label spreading algorithms. We also aim to assess the extent we *preserve precision* and how well the algorithm performs under scenarios of *limited labelled data*. We conclude this section with challenges faced by the label spreading algorithm presented by Zhou et al. [89] and describe how we provide novel additions to mitigate the challenges. To this end, we use the same performance metrics as in Chapter 4: *precision*, *recall* and $F_1$ *score*.

### 5.3.1 High-Recall

**Graph Setup.** In order to run the label propagation algorithm we construct an undirected adjacency matrix, initialised with all zeros which has entry $\mathbf{A}_{ij} = 1$ if content creator with id $i$ follows content creator with id $j$ or $j$ follows $i$ and $\mathbf{A}_{ij} = 2$ if they both follow each-other.

**Data Setup.** Here we evaluate the algorithm spreading 50% of the labelled content curators in the graph in Figure 5.2, with the remainder of the 50% of the labelled content creators used to evaluate the performance. Specifically, in the initial label matrix, only 50% of the nodes have labels where the label matrix has $\mathbf{L}_{ij} = 1$ if content creator with id $i$ has label $j$ and 0 otherwise.

**Metrics.** Similar to the other sections, ground truth data enables us to use the precision, recall and $F_1$ metrics which align with the high-level goals discussed in the introduction.

#### 5.3.1.1 Results

**Recall.** As expected, the results in Table 5.1 this component have higher recall than those in previous components. Those topics with lower recall (e.g. Coffee) appear to have lower recall due to the multi-topic nature of the content creators in the topic. For example, coffee content creators in this dataset are mostly coffee shops which often produce content which includes: *Coffee* making, *Interior Design* of the coffee shop, and about the *Healthy Food* which is sold in the shop. This result surfaces an interesting observation about this algorithm: *it predicts new topics*. During future work we will explore means of extracting the top $n$ topics predicted by the label propagation algorithm in order for cases like the coffee shop above to be labelled as all potentially correct topics. In conclusion, this algorithm was able to spread the labels from the small amount of lablled content creators to a large proportion of the un-labelled content creators, achieving our goal of high-recall.

**Precision.** We are pleased to observe that a high-precision is also preserved. This verifies the quality and effectiveness of our central hypothesis of this work: *connected content creators are likely to share similar topics*. Specifically, this shows that our hypothesis provides a topical signal with very little noise.

**Connectedness of Nodes.** Through exploration we observe that the topics with low $F_1$ score often have a low degree (poor connectedness), which limits the algorithms effectiveness in spreading labels accurately. For instance, with highly related topics like *Interior Designers* and *Ceramic*, with poorly connected *Interior Designers*, we may end up spreading ceramics to interior designers due to the imbalance of labels. We refer this issue as *graph sparsity*.

### 5.3.2 Robustness

To explore the semi-supervised LP algorithm's effectiveness in scenarios of poor supply of labelled data we compare the algorithm against the supervised bio text

classification we built in Chapter 4. In this experiment we trained the same models with varying training data size. Figure 5.3 shows the graph of these results, where we start with training the models with 80% of the labelled data and incrementally decrease it until 20% of the labelled data is used during training. It can be observed that the supervised text classification models $F_1$ *score* drops significantly, whereas the semi-supervised LS's score is only slightly affected. This demonstrates that using a semi supervised method is particularly suitable in our scenario of classifying topics with only a small number of labelled data points.

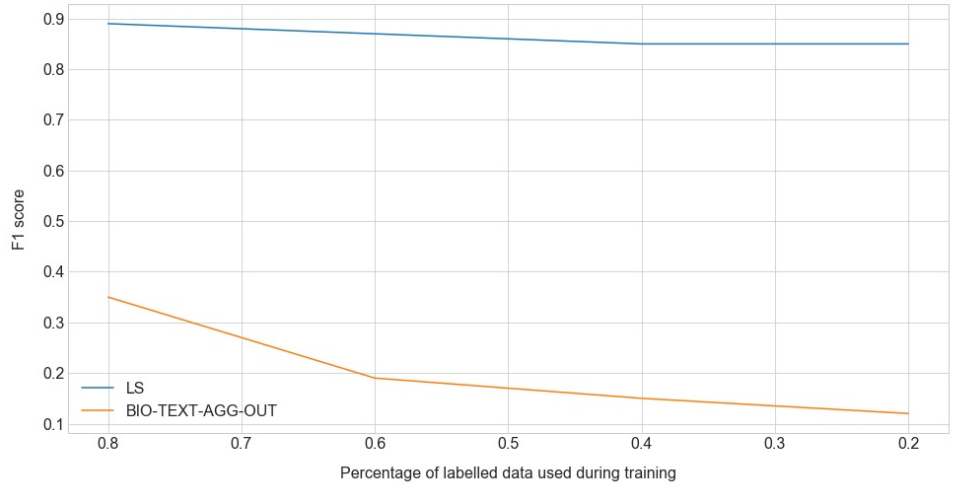



Figure 5.3: The $F_1$ score of label spreading (LP) and a text classification model (BIO-TEXT-AGG-OUT) as a function of the the amount of labelled data provided to the models.

## 5.4 Our Label Spreading Algorithm

In the previous section we observed at multiple stages the issue of **graph sparsity**, which leads to lower precision and recall with some topics. Additionally, through further visualisations we realised that there are some content creators who are poorly connected and follow other content creators out of interest. For example, a *Portrait Photographer*; may be poorly connected in the graph and follow a *Coffee* content creator out of interest. This would lead to the algorithm spreading the coffee label to a portrait photographer we call this problem **dissimilar label spreading**. In order to mitigate both issues of *graph sparsity* and *dissimilar label spreading* we propose a modification to the label propagation algorithm utilising topic similarity and multiple graphs of content creators.

### 5.4.1 Insights

**Tagged Users.** In Chapter 3 during the construction of our taxonoy we generated documents to represent communities of content creators. We observed that many content creators within topically coherent communities often tagged each other and

the same users. For example, many content creators in the van life community in Figure 3.7 frequently tagged the same van life accounts including @vanlifers, @vanlifediaries and @vanlifemovement with frequencies 34, 34, 29 respectively. We utilise this observation to make the first design choice:

**Design Choice 6.** *Use the highest tagged users of the labelled content creators to act as "anchors" between poorly connected content creators and their respective strongly connected topical cluster.*

With this design choice we form a series of hypothesise outlined below:

**Hypothesis 5.** *Design Choice 6 will enable our algorithm to spread labels through the topical tagged users, mitigating graph sparsity.*

**Hypothesis 6.** *Design Choice 6 will provide additional structure to the graph, allowing more distinct topical clusters to form, for which labels will be spread across.*

To provide further evidence as to the validity of this argument, we attempt to reason about these hypotheses through creating a visualisation of the same content creator graph in Figure 5.2 but with the popular tagged users added. In Figure 5.4 we present the graph with popular tagged users. We constructed the graph by first adding the tagged users who were tagged by at least 35% of the labelled content creators for each topic. Secondly, we added an edge between all labelled and unlabelled content creators and the newly added tagged users.

**Observation 1.** It can be observed that the communities in the new graph appear to be better separated, in particular, the *baking* and *healthy food* are a lot more distinct than in the previous follower-only graph. This provides evidence to support Hypothesis 6.

**Observation 2.** Additionally we note that there are nodes which provide topically degenerate connections, linking topically distinct communities. For example in Figure 5.4 we draw the readers attention to the large node highlighted by the arrow. The node is part of the (dark blue) community of coffee content creators and is connecting both the coffee and portrait photography clusters. The communities are topically dis-similar so we refer to this connection as a degenerate connection/bridge. Degenerate bridges cause the issue of *dissimilar label spreading* described earlier. The highlighted nodes size indicates that it has a large degree, hence is likely to be a "famous" user. This validates our design choice to remove nodes within the top $x$ degree percentile. To further mitigate the issue of spreading labels through degenerate bridges we make the next design choice of our label spreading algorithm:

**Design Choice 7.** *Labels should not be spread between two nodes who have dissimilar label distributions.*

For example, the probability of "Baking" and "Healthy Food" to be co-assigned to the same content creator should be larger than the probability of "Portrait Photography" and "Baking" to be co-assigned.

### 5.4.2 Algorithm Construction

Motivated by the insights in the previous sections we adapt Zhou et al. [89] regularisation function to ensure that only coherent labels are spread, thereby reducing *dissimilar label spreading*.

**Original Algorithm.** The regularisation framework/objective function presented by Zhou et al. [89] in Section 5.1.8 consists of a loss function which encodes the *smoothness constraint*: nearby nodes should have similar labels and the *initial labelling constraint*: initial label assignments should not change much.

**Setup.** We begin by defining two affinity matrices: a content creator graph $W \in \mathbb{R}^{N \times N}$ matrix where $W_{i,j} = 1$ if a content creator $i$ follows $j$. The topic graph $T \in \mathbb{R}^{L \times L}$ matrix which defines $T_{i,j}$ as the similarity between topics $i$ and $j$. Topic similarity is determined using Sematch[5], which computes the semantic similarity between concepts. Each topic in our taxonomy has a corresponding DBPedia page which can be used in the semantic similarity computation. Let $F = (F_1, .., F_N)^T = (C_1, ..., C_L)$ be an $N \times L$ matrix denoting the final topics assignments for each content creator, for example $F_{ij}$ represents the probability of content creator $i$ being assigned topic $j$. Let $Y = (Y_1, ..., Y_N)^T$ be an $N \times L$ matrix representing the initial topic assignments. Below we present the original regularisation framework proposed by Zhou et al. [89]:

$$\Omega(F) = \frac{1}{2}\Bigg( \underbrace{\sum_{i,j=1}^{n} W_{ij} \left\| \frac{F_i}{\sqrt{D_{ii}}} - \frac{F_j}{\sqrt{D_{jj}}} \right\|^2}_{\textit{Smoothness constraint on users}} + \mu \underbrace{\sum_{i=1}^{n} \|F_i - Y_i\|^2}_{\textit{Initial labelling constraint}} \Bigg) \tag{5.7}$$

where $\mu$ is the regularisation parameter and $Y^{(0)}$ is the initial label assignment matrix.

**Label Smoothing.** In order to utilise Design Choice 7 we add another regularisation term to Equation 5.7 to encode topic similarity. We refer to this as the *Smoothness constraint on topics*, this ensures that if two topics have high similarity, then they are likely to be assigned together for the same content creator. We can express this formally by stating that the columns of $F$, $(C_1, ..., C_L)$ which represent the vectors of each topic should be similar if the topics are similar. This condition is added to the original regularisation framework:

[5] https://github.com/gsi-upm/sematch

$$\Omega(F) = \frac{1}{2}\Bigg( \underbrace{\sum_{i,j=1}^{n} W_{ij} \left\| \frac{F_i}{\sqrt{D_{ii}}} - \frac{F_j}{\sqrt{D_{jj}}} \right\|^2}_{\textit{Smoothness constraint on users}} \tag{5.8}$$

$$+ \frac{1}{2}\mu \underbrace{\sum_{i,j=1}^{n} T_{ij} \left\| \frac{C_i}{\sqrt{D'_{ii}}} - \frac{C_j}{\sqrt{D'_{jj}}} \right\|^2}_{\textit{Smoothness constraint on topics}} \Bigg) \tag{5.9}$$

$$+ \underbrace{\eta \sum_{i=1}^{n} \|F_i - Y_i\|^2}_{\textit{Initial labelling constraint}} \tag{5.10}$$

Where $D$ and $D'$ are the diagonal matrices where $D_{ii} = \sum_{j=1}^{N} W_{ij}$ and $D_{ii} = \sum_{j=1}^{N} W'_{ij}$. The hyperparamters $\mu$ and $\eta$ control how much weighting the regularisation constraints in the formulation of $F$. The first term in Equation 5.8 can be re-written into matrix form as demonstrated below:

$$\begin{aligned}
\frac{1}{2}\sum_{i,j=1}^{N} W_{ij} \left\| \frac{F_i}{\sqrt{D_{ii}}} - \frac{F_j}{\sqrt{D_{jj}}} \right\|^2 &= \frac{1}{2}\Big( \sum_{i=1}^{N} F_i^T F_i + \sum_{j=1}^{N} F_j^T F_j - 2\sum_{i,j=1}^{N} \frac{W_{ij} F_i^T F_j}{\sqrt{D_i D_j}} \Big) \\
&= \sum_{i}^{T} F_i - \sum_{i,j=1}^{N} W_{ij} \frac{F_i^T F_j}{\sqrt{D_i D_j}} \\
&= tr(F^T (I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}) F) \qquad = tr(F^T L F)
\end{aligned}$$

where $L = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$ i the *Normalised Laplacian* of the graph associated with $W$. The second term in Equation 5.8 can be re-written similarly, providing a Equation 5.8 in matrix form as follows:

$$\Omega(F) = tr(F^T L_u F) + \mu tr(F L_l F^T) + \eta \|F - Y\|^2 \tag{5.11}$$

where $L_u = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$ and $L_t = I - D'^{-\frac{1}{2}} T D'^{-\frac{1}{2}}$ are the normalised Laplacians of the user graph and label graph respectively. Now the most optimal label assignment matrix is

$$F^\star = \arg\min_F \Omega(F) \tag{5.12}$$

To determine the optimal we find the stationary point of $\Omega$. By using the following matrix properties below

$$\frac{\partial tr(X^T A X)}{\partial X} = (A + A^T) X \tag{5.13}$$

$$\frac{\partial tr(X A X^T)}{\partial X} = X(A + A^T) \tag{5.14}$$

and noting that $L_u$ and $L_t$ are symmetric matrices we are able to find the derivative of $\Omega$, hence the optimal topic assignment matrix $F^\star$:

$$\left.\frac{\partial \Omega}{\partial F}\right|_{F=F^\star} = L_u F^\star + \mu F^\star L_l + \eta(F^\star - Y) \tag{5.15}$$

$$= 0 \tag{5.16}$$

Which can be re-written as:

$$(L_u + \eta I)F^\star + \mu F^\star L_l = \mu Y \tag{5.17}$$

that can be solved using numerical or iterative solutions of matrices in the form $AX + XB = C$, where $A \in \mathcal{R}^{m \times m}, B \in \mathcal{R}^{n \times n}$ and $X, C \in \mathcal{R}^{m \times n}$.

### 5.4.3 Evaluation of Our Label Spreading Algorithm

**Graph Setup.** We construct affinity matrices $W$ and $T$ as described in Section 5.4.2 and construct an initial label matrix with 50% of the same dataset as in Section 5.3.1. Additionally we add the users tagged by at least 35% of the labelled content creators for each topic. Secondly, we added an edge between all labelled and un-labelled content creators and the newly added tagged users.

**Data Setup.** Here we evaluate the algorithm spreading 50% of the labelled content curators in the graph in Figure 5.2, with the remainder of the 50% of the labelled content creators used to evaluate the performance. Specifically, in the initial label matrix, only 50% of the nodes have labels where the label matrix has $\mathbf{L}_{ij} = 1$ if content creator with id $i$ has label $j$ and 0 otherwise.

**Metrics.** Similar to the other sections, ground truth data enables us to use the precision, recall and $F_1$ metrics which align with the high-level goals discussed in the introduction.

**Baselines.** In order to assess how well our algorithm solved the issues we discussed previously we compare it to the standard label spreading algorithm evaluated in Section 5.3.1, we call this `LS` and ours `TS-LS` for *Topical-Similarity Label Spreading*.

**Results.** From Table 5.2 we can confirm that the design decisions made in the previous sections indeed improve the algorithm by, for both precision (an increase in 0.07) and recall (an increase in 0.06).

## 5.5 Future Work

### 5.5.1 Inductive Topic Inference

The label spreading algorithms we presented this chapter are *transductive*, whereby they optimise an objective function to find the labels of the existing unlabelled data

using the labelled and unlabelled data. These transductive algorithms do not have the ability to handle new data. To handle new content creators we could use K-NN methods whereby one would predict the labels of a new content creator by averaging the labels of it's nearest neighbours. This method lacks the global information that label spreading utilises. In order for our algorithms to perform *inductive label/topic inference* i.e. handling new content creators we aim to explore inductive algorithms that utilise global information of the graph with the regularisation conditions we introduced in our TS-LS algorithm. We start future work with the minimal optimisation function defined in Equation 5.18 and continue to add regularisation terms to improve the performance. The equation below aims to determine the new label $\hat{y}$ by minimising the objective function, where $N$ represents the number of content creators in the current graph and $W_k$ indicates the adjacency matrix of the content creator graph with the new node $k$ included.

$$\Omega(\hat{y}) = \sum_{j}^{N} W_{ij}^{k}(\hat{y} - \hat{y}_j)^2 + \mu\hat{y}^2 \tag{5.18}$$

Table 5.1: Precision, recall and $F_1$ score for each topic achieved by the label spreading algorithm.

| Topic | Precision | Recall | $F_1$ score |
|---|---|---|---|
| Skateboarding | 0.98 | 0.95 | 0.96 |
| Nature Photography | 0.89 | 0.76 | 0.82 |
| Dogs | 0.93 | 0.74 | 0.82 |
| Tattoo | 0.87 | 0.98 | 0.92 |
| Baking | 1.00 | 0.80 | 0.89 |
| Interior Design | 1.00 | 0.40 | 0.57 |
| Painting | 0.83 | 0.86 | 0.85 |
| Ceramics | 0.80 | 0.90 | 0.85 |
| Portrait Photography | 0.81 | 0.62 | 0.70 |
| Surfing and Bodyboarding | 0.90 | 1.00 | 0.95 |
| Skiing | 0.93 | 1.00 | 0.96 |
| Plant Based | 0.71 | 0.76 | 0.74 |
| Wedding Photography | 1.00 | 0.67 | 0.80 |
| Climbing | 0.83 | 0.96 | 0.89 |
| Ocean | 0.71 | 1.00 | 0.83 |
| Coffee | 0.86 | 0.67 | 0.75 |
| Supercars | 0.93 | 0.96 | 0.94 |
| avg | 0.87 | 0.86 | 0.86 |

Table 5.2: Precision, recall and $F_1$ score achieved by the original label spreading algorithm (`LS`) and our modification (`TS-LS`)

| Algorithm | Precision | Recall | $F_1$ score |
|---|---|---|---|
| `LS` | 0.87 | 0.86 | 0.86 |
| `TS-LS` | 0.94 | 0.92 | 0.93 |

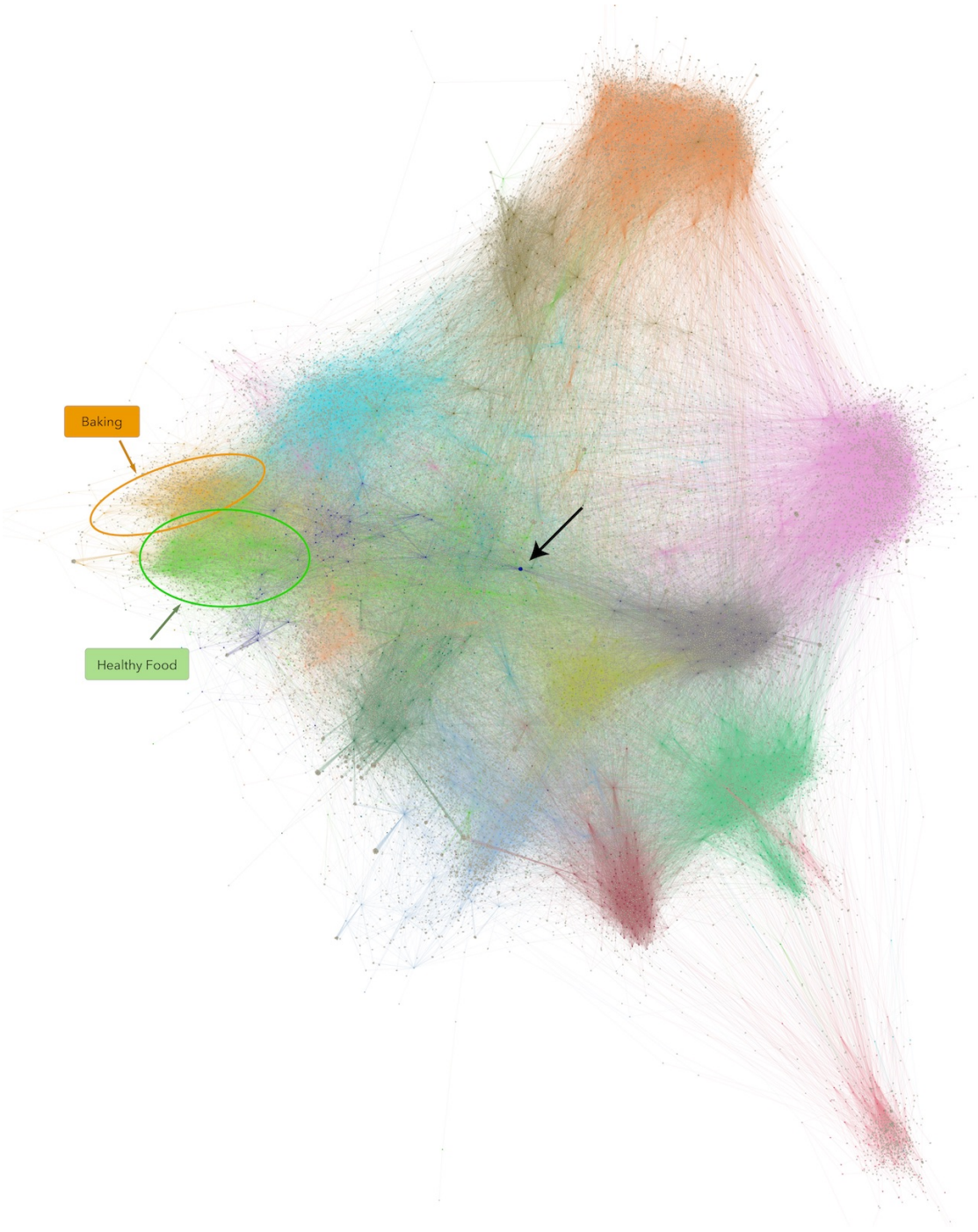



Figure 5.4: A topical community graph of influencers as nodes and the edges between representing the "following" relation and the "tagged" relation, visualised with a Force Layout [23]. The graph was generated by starting with a small number of labelled influencers and adding the influencers in their neighbourhoods. This visualisation shows each topic as a separate colour, with influencers and their edges coloured by their labelled topic. A nodes size is reflective of its degree[4]. The large proportion of the graph is *unlabelled*, with the colour grey.

# 6 Data Collection and Storage

In this chapter we provide an overview of how we collected the data used in this work. Additionally we describe how we utilised polygot persistence to effectively store the data such the data storage technology is best chosen to satisfy our applications demands.

## 6.1 Polygot Persistence

Polygot Persistence refers to the storing of data in multiple data storage technologies such that each is chosen to fit the needs of individual components/applications of a system. In this section we breifly describe the various data storage technologies in our architecture (Figure 6.1).

### 6.1.1 Document Store: MongoDB

MongoDB is a document store which holds data in flexible, JSON-like documents, meaning fields can vary from document to document and data structure can be changed over time[1]. It is suitable for us to store the data from our data collectors for the following reasons:

| | |
|---|---|
| *Evolving Schema.* | MongoDB's schema-less nature is well suited to the dynamic and inconsistent nature of the data we're collecting. |
| *Horizontal Scaling.* | As our data grows, MongoDB replica sets makes it very easy for us to scale horizontally, adding more power and storage when needed. |
| *Data Collection Format.* | MongoDB is well suited to deal with the data format (JSON) arriving into it from our data collection applications. |

### 6.1.2 Graph Database: Neo4j

Our work heavily relies on fast graph queries (most frequent of which are neighbourhood queries and random walks). We didn't want to force MongoDB (a document-based data store) to handle these graph-style relationships because the implementation would have been costly and inefficient. Instead, we use a polyglot persistence approach to capitalise on the strengths of each, deciding to use both MongoDB and Neo4j together.

[1] https://www.mongodb.com/what-is-mongodb

### 6.1.3 Full Text Search: Elasticsearch

Elasticsearch is a distributed search engine based on Lucene. It provides a distributed, multitenant-capable full-text search engine with an HTTP web interface and schema-free JSON documents. Its JSON like storage and fast full text search make it a particularly good choice to store our streaming data to be searched by the end user.

## 6.2 Data Collection

### 6.2.1 Overview and Sampling Strategy

**Challenges.** The aim is to collect the profile data and graph data of *content creators* with minimal amount of HTTP requests to Instagram API. We aim to limit the amount of requests because (i) we are limited by Instagram to a certain rate for which we are able to send requests and (ii) to decrease the time required to collect the large amount of data we require.

**Solution.** In order to collect the graph data from Instagram that this work required we undertook a *biased snowball sampling* approach. Our approach starts with a seed node (user) $s$ and collects data from Instagram API (Section 6.2.3) as described in Algorithm 11. The algorithm we designed is biased by prioritising data collection of highly-engaged users, improving the likelihood of spending expensive API requests on content creators.

**Algorithm 11** Biased Snowball Sampling Algorithm

**Input:** Initial seed $s$
**Output:** Graph and profile data of a sample of Instagram content creators

```
Initialise priority queue Q = {s} ordered by engagement (Equation 3.24)
while |Q| is non-empty do
    s ← pop(Q)
    F^s_out ← all users which s follows collected with API Endpoint 1
    for u in F^s_out do
        d_u ← profile data collected using API Endpoint 2
        if d_u indicates that u is content creator then
            Save d_u in Instagram profile MongoDB collection
            p_u ← most recent 14 posts collected from API Endpoint 3
            Store posts p_u in Instagram media collection
            Calculate engagement and add to Q
        end if
    end for
end while
```

**Result.** Collected $2,687,894$ content creators and $101,747,430$ Instagram posts over a two month period.

### 6.2.2 Distributed Queues

In order to achieve the result described above (Section 6.2.1) we made various design choices for our distributed data collection architecture. As the design of the tech stack is not the focus of this work, we provide in this section only a brief overview of our design choices and reasoning where appropriate.

**Redis Backed Queue.** To enable us to distribute our data collectors over a cluster of servers, we had the challenge of enabling communication between them. This communication was necessary for all data collectors to have access to the same priority queue described in Algorithm 11. To achieve this we created a server to host a distributed queue built with well-maintained Node.js queue, *BullJS*[2] backed by a Redis server. It is important to note that all components of our data collection and data storage architecture were deployed to a VPC[3], such that no external entities could interfere with the internal services.

**MongoDB Replica Set.** To handle the large output from the data collectors we deployed a replica set of MongoDB instances which provided redundancy and high availability.

**AWS Spot Instances.** To deploy the data collection servers we used AWS spot instances [4]. This provided a significantly more economical way to run our data collectors than on-demand instances. We typically scaled up the number of instances in our fleet when the cost was low. Practically, we created a custom Amazon AMIs[5] holding the code to ensure that a new spot instance correctly downloaded and run the latest version of our data collection application.

**Node.js.** A profitable language choice for building our data collectors was JavaScript run in Node.js[6] run-time. This allowed us to fully utilise the power of network programming with Node.js' event-driven, non-blocking I/O model. Node.js, built to handle async I/O from the ground up made it easy for us to achieve exceptional performance when making API requests to Instagram and the wide array microservices in our architecture.

### 6.2.3 Instagram API

We created various custom libraries which, when given data from the Instagram API, formatted it then inserted it into MongoDB. The API endpoints we used are summarised below. These libraries also handle adding new requests to the queue described in Section 6.2.2 and determining whether a user is a content creator or a normal/passive user. As data-collection is not the focus of this work we leave out the details of what data is returned by the API's and how we processed it.

**API Endpoint 1** (*/users/{user-id}/follows*)**.** *Provides a paged response for the users for which* `{user-id}` *follows.*

[2] https://github.com/OptimalBits/bull
[3] https://aws.amazon.com/vpc/
[4] https://aws.amazon.com/ec2/spot/
[5] https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AMIs.html
[6] https://nodejs.org/en/

**API Endpoint 2** (*/users/{user-id}*)**.** *Provides basic profile information of* `{user-id}`*, for instance a users biography, full name and number of followers.*

**API Endpoint 3** (*/users/{user-id}/media/recent*)**.** *This endpoint provides the recent media posted by* `{user-id}` *including information like the number of comments, the number of likes, caption and users tagged in the post.*

### 6.2.4 Streaming from MongoDB into Neo4j

The goal of this section is to outline how we synchronise a subset of data from MongoDB to Neo4j. This allows us to stream data as it arrives to MongoDB into Neo4j, keeping all our data stores in sync.

**Streaming from MongoDB's Oplog.** MongoDB provides a library which is perfect for this use case: *mongo-connector*[7], this provides us with a means to listen for all update operations in MongoDB and mirror them to other systems. *mongo-connector* ensures that data is consistent across all data sources by tailing the MongoDB oplog. This library provides a means to synchronise data with Elasticsearch and Solr.

**Mapping data from MongoDB to Neo4j.** To facilitate synchronising data from MongoDB to Neo4j we use *Neo4j Doc Manager*[8]. This is built upon *mongo-connector* and allows developers to create mappings between MongoDB and Neo4j. Mappings need to be defined as documents stored in MongoDB are in a similar form to JSON whereas the same data in Neo4j are stored as graphs.

**Example.** Figure 6.2 shows a sample of how documents in MongoDB are mapped to Neo4j graph format. The Instagram profile data in the sample document in Figure 6.2a mapps to a *node* (nk7) and stores the relevant properties within it. Additionally, the same document has a list of users for which nk7 follows, our custom mapping uses these to create *relationships* between the relevant nodes, for instance it will create a `FOLLOWS` relationship between *nk7* and *tong.my*. The document in Figure 6.2b represents a post of nk7s, our mapper uses this information to create the `TAGGED` relationship in Neo4j, indicating that *nk7* tagged *blackbab*. If the tagged relationship already exists (i.e. nk7 has tagged blackbab before) then the weight of the relationship is incremented by one.

[7] https://github.com/mongodb-labs/mongo-connector
[8] https://github.com/neo4j-contrib/neo4j_doc_manager

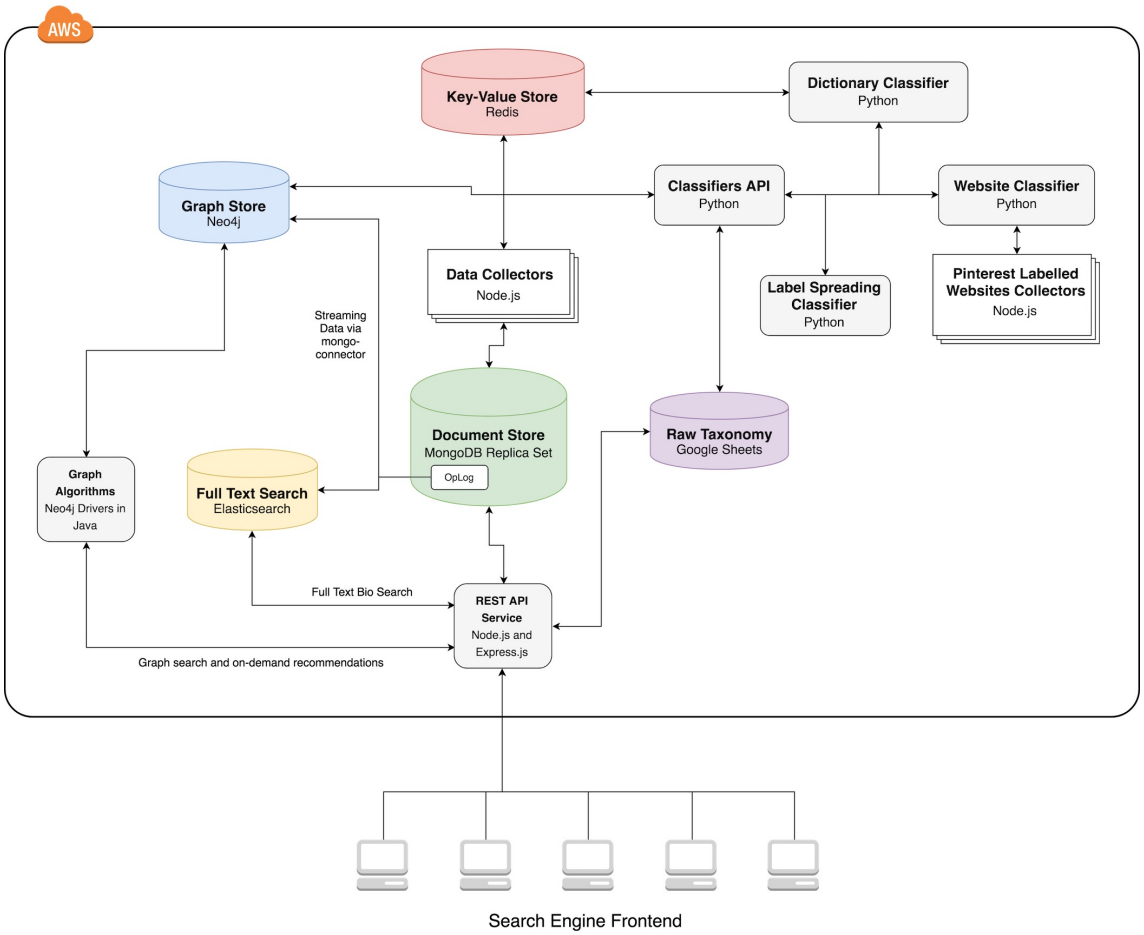


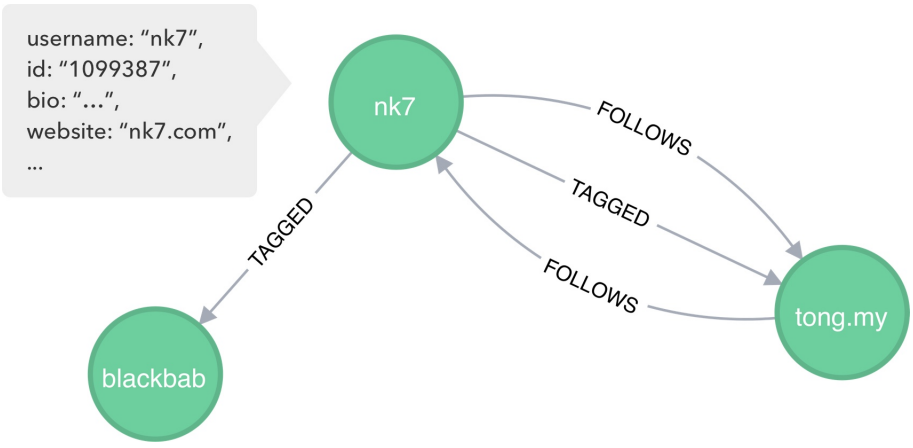

Figure 6.1: The main components of our architecture and applications.

```
{
  "id": "1099387",
  "username": "nk7",
  "profilePicture": "https://...",
  "fullName": "NKCHU",
  "bio": "nk.7@outlook.com ...",
  "website": "nk7.com",
  "lastUpdated": {
    "$date": "2018-04-26T21:53:22.776Z"
  },
  "following": [
    {
      "id": "620226",
      "username": "tong.my"
    },
    ...
  ],
  "counts": {
    "media": 1617,
    "follows": 168,
    ...
  },
  ...
}
```


(a) Example of a BSON document from the MongoDB Instagram collection


```
{
  "ownerId" : "1099387",
  "username" : "nk7",
  "mediaId" : "1604362831519925839_1099387",
  "link" : "https://...",
  "caption" : "wandering around the bund.",
  "images" : [
    {
      "url" : "https://...",
      "height" : 1349,
      "width" : 1080
    },
    ...
  ],
  "counts" : {
        "likes" : 5843,
        "comments" : 68,
        ...
  },
  "usersTagged" : [{
      "username" : "blackbab",
      ...
  }],
  ...
}
```


(b) Example of a BSON document from the MongoDB InstagramMedia collection




(c) The nodes and relationships created in Neo4j from the MongoDB documents in Figures 6.2a and 6.2b.

Figure 6.2: Data stored across MongoDB and Neo4j

# 7 User Applications

## 7.1 Architecture

Modularity is key for any multi-purpose architecture. For this reason we designed our architecture and the environment surrounding it using a *Microservices Architecture*. Each component of our architecture is implemented as a separate microservice which has a well defined API which it must conform to. For instance, our dictionary classifier is deployed separately to our website classifier, where they both have exposed endpoints which can be queried via a HTTPS REST API. The design details of our REST api are left out here for the sake of brevity. The overall design of our architecture is described in Figure 6.1 where all of our services are deployed within a VPC[1] on AWS[2]. Two core benefits of designing it in this way is are:

**Flexibility.** If we were to change the way our dictionary classifier performs it's classification in a fundamental way, then as long as it still conformed to the API specification then no other components would have to change.

**Scalability.** The distributed and replicated manner of our architecture design allows us to scale each component as user demand increases. For instance, if the demand on Elasticsearch is increased we are able to simply add a new node to our cluster.

The data collection and storage is described in Chapter 6. Here we focus on the design of our front end and back end of the search engine. The front end application (search engine) is able to communicate with our Express API, which exposes an endpoint which queries our Elasticsearch cluster. Note that our application is currently closed for public so requires correct login details to access the search engine. We secure our applications by following this flow of authentication, whereby a user must own a fresh access token to request our API server.

[1] https://aws.amazon.com/vpc/
[2] https://aws.amazon.com/

## 7.2 Search Engine Implementation

### 7.2.1 React JS

A natural choice for the front end library for our search engine was React JS[3]. Below we outline two of the most appealing features of this library:

1. **Component First.** React allows for the creation and use of re-usable, composable and stateful components. With a rich developer community, this made it extremely easy for us to incorporate open-source well-tested libraries, additionally it allowed us to re-use components across our application(s). Furthermore it provided a means for fast and consistent software engineering.

2. **Virtual DOM.** Previous to React, interactive applications that required manipulation of the DOM was very hard to do in a performant manner, usually leading to sluggish user experiences with large code-bases for seemingly simple and small applications. React solves this issue by providing a virtual DOM that lives in memory. The developer directly interacts with the virtual DOM and React automatically determines the most efficient way to apply the minimal number of changes to the real DOM. This provides an excellent user experience and effortless development experience.

This library provides functionality to build front end components, however has limited out-of-the box design and functionality to build scalable web applications which communicate with various APIs asynchronously. For this reason we utilised various other libraries to assist in other parts of web development. An overview of our front end design is inspired by a scalable open-source front end development boilerplate[4]. We provided various modifications to this boilerplate for it to suit our specific needs. These include (i) auth flow, allowing users to log in with their Google emails (only verified emails are able to log in successfully) (ii) various theming modification, among other modifications. An overview of the communication between various components of our front end is described in Figure 7.1. We refrain from describing details of the frontend implementation as this is not the focus of this work, however we encourage an interested reader to visit the boilerplates readme[5] to understand how one could build a scalable front end application.

### 7.2.2 Express API

The responsibility of this NodeJS REST API server is intended to expose search functionalities to our front end application. Our server was built with various popular design principles in mind. This server handles parsing requests from the front end, forwarding them in the correct format to services like ElasticSearch and Neo4j and finally returning the results to the front end in a consistent form. In addition, our server handles all errors in a deterministic manner to provide useful reporting back to the user-facing application. Finally, this server is also responsible for

[3] https://reactjs.org/
[4] https://github.com/react-boilerplate/react-boilerplate
[5] https://github.com/react-boilerplate/react-boilerplate/blob/master/docs/

handling feedback from the user application about the correctness of our classifications. It processes feedback requests from the front end and stores the results in a MongoDB collection to be used to improve the quality of our classifiers.

## 7.3 Recommendation System

This section briefly describes the design and development of a proof-of-concept recommender system which uses the classifications made by our architecture. Given a content creator as a query, it should provide the most **topically related content creators** from millions of content creators in real time.

### 7.3.1 Design Choices

We start with outlining the key requirements we aim for our algorithm to satisfy.

**Topical Relevance.** The recommendations should not be biased by age, gender, posting behaviours or any other user/engagement features. With the goal of providing better topical recommendations than the suggested feature on Instagram. Additionally this will help prevent an *algorithmic glass ceiling* to be formed, as described by Stoica et al. [75]. This leads to our first design choice

**Design Choice 8.** *Recommendations are solely based on the topics of content creators.*

In future work we aim to extend the features we use to recommend in order to improve topical recommendations.

**Real Time.** The recommendations should happen in real-time and as quickly as possible, i.e. utilise the most up to date information. This is key as our architecture is fairly dynamic, whereby new topics are added to our taxonomy regularly, hence providing more detailed topical classifications of content creators. Our requirement of real-time recommendations leads to our second design choice:

**Design Choice 9.** *We will not use recommendation algorithms which pre-compute recommendations, instead we perform real-time graph traversal algorithms which are independent of graph size by utilising Neo4j graph storage.*

### 7.3.2 Algorithm

In our proof-of-concept algorithm we design it in a similar way to the Pixie Random Walk algorithm presented by Eksombatchai et al. [19] and used in Pinterest to recommend pins to users. We provide an overview of the features the algorithm inspired by Eksombatchai et al. [19] below:

#### 7.3.2.1 Biased Random Walks

Pixie performs biased random walks biased in a user-specific way. From Design Choice 8 we aim to bias our random walks on the topics of content creators. To

perform a biased random walk from node $i$ we pick the node $j$ in $i$'s neighbours that has the largest cosine similarity between $i$ and $j$'s topic vector. The topic vector of a node $i$ is the row $F_i$ in the topic assignment matrix $F$ computed by our label spreading algorithm. This provides high quality topic vectors which not only provide information about a content creators predominant topic but the probability distribution over all topics.

**Implementation.** We implement are algorithms in Java due to extensive open-source community around Neo4j algorithms in Java. We used Neo4j Driver[6], which launches an embedded database allowing for low latency queries to the graph.

Algorithm 12 describes how we arrive at a set of recommendations from a query content creator. The visit counts represent the number of times a content creator is visited on a random walk, the higher the visit count, the more relevant a recommended content creator is. The function `PickBiasedNode`$(N, c)$ picks a node $c'$ in $c$'s neighbourhood such that $c$ and $c'$ have the largest cosine similarity between their topic vectors. `SampleWalkLength`$(\alpha)$ samples a walk length depending on the parameter $\alpha$.

**Algorithm 12** Topical Random Walk

**Input:** Query content creator $q$, nodes $N$, node-topics probability matrix $F$, number of random walk steps $N$, number of random walks $\alpha$
**Output:** $V$ visit counts of recommended content creators

$V \leftarrow \emptyset$
$steps \leftarrow 0$
**while** $steps \leq N$ **do**
  $c \leftarrow q$
  $rwLen \leftarrow$ `SampleWalkLength`$(\alpha)$
  **for** $i$ **to** $rwLen$ **do**
    $c' \leftarrow$ `PickBiasedNode`$(N, c)$
    $V[c'] + +$
    $c \leftarrow c'$
  **end for**
  $steps \leftarrow steps + rwLen$
**end while**

### 7.3.3 Evaluation

Due to time restrictions we were unable to perform rigorous evaluation on our recommendation system. In future work we aim to aim to extend our recommendation system and perform comprehensive evaluation. For the time being, we perform a subjective comparison of our recommendation system against Instagram's suggested feature. Tables 7.1, 7.2 and 7.3 show that our recommendation system consistently provides topically relevant recommendations whereas Instagram's suggested feature seems to be biased to recommending friends of the query content creator or other "famous" accounts.

[6] https://github.com/neo4j/neo4j-java-driver

Table 7.1: Subjectively comparing our recommendation system to Instagram's suggestion feature. Each content creator is represented by a (username, topic) pair. We are aiming to show that our recommendation system makes more topical recommendations than Instagram's suggestion feature. The query content creator in this table is **@_shootthephoto_**, a **Sports Photographer**

| Instagram's Recommendations | | Our Recommendations | |
|---|---|---|---|
| username | topic | username | topic |
| vikkstagram | Gaming | trailjunkiephotos | Sports Photography |
| behzingagram | Gaming | bystith | Football Photography |
| sidemenclothing | Merchandise Company | erikwestbergphotography | Outdoor Sports Photography |
| realcalfreezy | Gaming | redbullillume | Sports Photography |
| pureminterr | Gaming Fan Account | jeffdivinephotographer | Surf Photography |

### 7.3.4 Future Work

#### 7.3.4.1 User-Content Creator Recommendations.

We aim to extend our recommendation system to recommend topical content creators to a user. This would manifest in a similar way to our current algorithm except it would perform graph traversal on a *bipartite graph* connecting the content creators graph and user graph through various relationships, e.g. a user in the user graph may follow $x$ users in the content creator graph, thereby connecting the two graphs with the following relation.

#### 7.3.4.2 Multiple Weighted Content Creator Queries

We aim to extend this algorithm by incorporating the idea of multiple weighted query content creators used in Pixie [19]. A weighted query set is a set of content creators that a user follows, as opposed to a query containing just one content creator creator. The weighting of a content creator $c$ can be determined by how much a user has recently engaged with $c$. Using this query set we can perform multiple random walks from each content creator in the query set and use the combined visit counts in order to make more relevant recommendations. A candidate content creator is one which is found at the end of a random walk, when a candidate is reached we consider it *visited*. Those candidates which are reached by many random walks by many independent queries (within the same query set) are considered more relevant. To encode this one could combine the visit counts from multiple queries using the following equation proposed by [19]:

Table 7.2: Subjectively comparing our recommendation system to Instagram's suggestion feature. Each content creator is represented by a (username, topic) pair. We are aiming to show that our recommendation system makes more topical recommendations than Instagram's suggestion feature. The query content creato's topic is **Skydiving** (**@ericpeterminze**)

| Instagram's Recommendations | | Our Recommendations | |
|---|---|---|---|
| username | topic | username | topic |
| madymorrison | Yoga | nicwildandfree | Skydiving |
| elizaxlara | Lifestyle | skydivids | Skydiving |
| ginastiebitzofficial | Model | majid_moshfeghi | Skydiving Pilot |
| berlinaaalter | Sports | skydivingposts | SkyDiving |

$$V[c] = \big(\sum_{q \in Q} \sqrt{(V_q[c])}\big) \tag{7.1}$$

Table 7.3: Subjectively comparing our recommendation system to Instagram's suggestion feature. Each content creator is represented by a (username, topic) pair. We are aiming to show that our recommendation system makes more topical recommendations than Instagram's suggestion feature. The query content creator in this table is **@jatiputra**, a **Graphic Designer**. This account is an example of where Instagram provides a limited amount of recommendations (3).

| Instagram's Recommendations | | Our Recommendations | |
|---|---|---|---|
| username | topic | username | topic |
| natgeotravel | Travel | graphicdesignersclub | Graphic Design |
| nasa | Space | douggraphics | Graphic Design |
| unsplash | Photography | daily_minimal | Minimal Graphic Design |
| - | - | douggraphics | Typography Graphic Design |
| - | - | razvanvezeteu | Graphic Design |

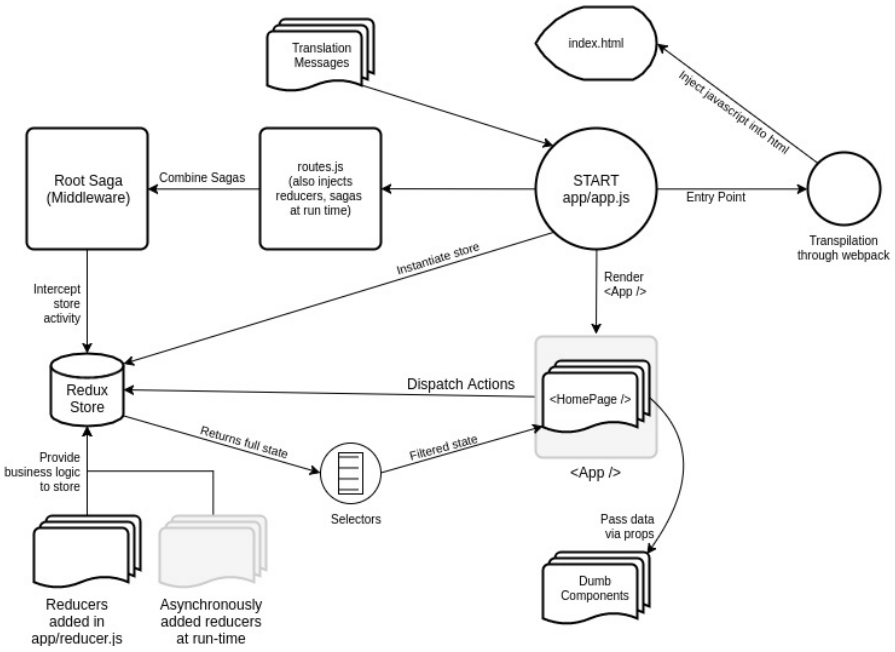



Figure 7.1: An overview of how the components of our frontend communicate. An interested reader can learn more by reading https://github.com/react-boilerplate/react-boilerplate/blob/master/docs/ docs

# End-User Application and Holistic Evaluation

# 8

## 8.1 Introduction

In this chapter we provide a holistic evaluation of our achievements. For the purpose of clarity and relevance we provide a detailed per-component evaluation at the end of each relevant chapter. To clearly describe an overview of our achievements, we re-iterate our goals and contributions from our introduction. Each goal has an associated unique identifier which we reference during the process of describing our holistic evaluation in this chapter. Our are classified into three categories: Research (**R**), Business/Societal (**B**) and Engineering (**E**) Contributions. Each begins with their unique identifier.

- `TAX` **B, R, E** A first attempt to constructing a taxonomy which captures a large portion of topical content on Instagram
- `T-LCD` **R** A novel algorithm topical local community detection algorithm for niche topical communities of content creators
- `CLF` **R** A first attempt to classify content creators on Instagram, with high-precision into a large taxonomy of potential topics
- `LBL-DATA` **E** A robust pipeline to automatically label content creators of high-precision topics with minimal supervision
- `TS-LS` **R** A novel modification of the label spreading algorithm that spreads topical labels across a graph of content creators
- `ARCH` **E** A scalable architecture to infer the topics of Instagram content creators, deployed in production and used in multiple real-world applications
- `SEARCH` **B, E** Develop and deploy a search engine allowing users to navigate the topical content on Instagram, specifically providing the first content creator search engine with niche topics in a variety of fields
- `RECOM` **B, E** A topical content creator recommendation system, allowing users and brands to find new high-quality content creators

## 8.2 The Narrative of Our Achievements

This work aimed to provide a first attempt to classify the niche topics of millions of content creators on Instagram. In order to achieve this, we split the core goal down into sub-problems which are each tackled by a separate component of our architecture. These sub-problems are outlined by asking a series of research questions. To ensure each component effectively solves its sub-problem, we perform extensive reasoning about the challenges each component might face and how we might mitigate them. Finally we conclude the construction of each component with a relevant evaluation showing how the component specifically solves the sub-problem it is constructed to solve. We started our research by observing that, in order to classify content creators by topic, we must outline what the topics are, hence we aimed to answer the first question:

> *How does one discover the niche topics that exist on Instagram and organise them in an interpretable manner?*

**Component 1: Taxonomy Creation via Topical Community Detection.** Through defining relevant evaluation metrics we were able to show that our topical local community detection algorithm (`T-LCD`) outperformed popular community detection algorithms in finding small, tightly knit topical communities. Specifically our method had 42% higher topical coherence than the next best alternative and an average community size of 86.5 which indicates smaller niche communities, in comparison to the general topical communities of the next best algorithm with an average size of 1973. In addition to the metrics we defined to evaluate our communities we provided visualisations which present the same results in a more intuitive manner, indicating that our communities are well separated, small and tightly connected. Additionally, we generated various documents to display the topical communities which were used by a team of Instagram marketing experts with the aim of *novel topic discovery*, through this procedure we were able to empirically validate the success of our T-LCD algorithm by discovering many niche topics which were previously undiscovered to extend our taxonomy. This resulted in a large (and growing) extension to our initial taxonomy (`TAX`) which was built using public taxonomies and knowledge bases. The goal of this component was to achieve *high-coverage* over topics, which is achieved through continuous utilisation of approaches specified in this component. Naturally, this led to the formation of our next question:

> *How can the content creators be classified into their topics of content with high-precision?*

**Component 2: High Precision Topic Inference Pipeline.** We recognised that traditional methods proposed in past literature couldn't answer this question effectively due to the unique challenges posed by Instagram, end-user requirements and the large taxonomy of topics. To this end, we determined the *most accurate topical signals* of an Instagram account and extended the ideas in past work to define a pipeline that infers the topics of content creators with *high-precision*. This pipeline was built upon novel hypotheses that consider the relationships between content creators which we showed through thorough evaluation to meet its goal of *high-precision* exceptionally well (`LBL-DATA`). Specifically our best model, attempting to

maximise precision through an ensemble of classifiers achieved a precision of 0.99 with an improvement of 0.56 precision against our baseline text classification model over a ground truth dataset (Section 4.7). At this point we observe that our ensemble classifier suffers from poor recall, and hence we face the challenge of answering the final question:

> *How do we classify the topics of the remainder of the content creators whilst preserving precision?*

We answered this question by combining the limited amount of labelled data that our pipeline acquired and richness of information in our content creator graph to build a graph based semi-supervised learning (GB-SSL) algorithm. Specifically we transform strong hypotheses and design choices into a concrete objective function which can be optimised to generate a labelling matrix that assigns content creators to a probability distribution over topics. We demonstrated experimentally that our GB-SSL algorithm (`TS-LS`) obtains an $F_1$ *score* 7% higher the standard approach of label propagation (Section 5.3.1). Finally we performed experiments comparing our algorithm against supervised methods showing that our method is significantly less susceptible to varying availability of labelled data (Section 5.3.2). Finally, we combined the components built into an architecture which is used to provide labelled content creators to our search engine for topical content creators on Instagram (`CLF`, `ARCH`).

## 8.3 Searching for Topical Content Creators

In order to to conclude our evaluation, we show how the architecture we developed in this work achieved our core motivation: *to provide a means to navigate the topical world of Instagram.* We demonstrate this by presenting a minimal viable version of a topical content creator search engine (`SEARCH`). We compliment our search with a proof-of-concept recommender system designed in Section 7.3, our algorithm performs topically biased random walks on the content creator graph with the aim of providing high-quality topical content creator recommendations from an initial query content creator (`RECOM`). In order to gain an initial assessment of the quality of our recommendations, we subjectively compare the recommendations made by our system to Instagram's suggested feature and found that the initial results show our system provides more topically relevant recommendations (Section 7.3.3). Figure 8.1 provides an example query to our search engine. In this example, a user is searching for *Minimal Artists* (red box). Notice that the results don't necessarily include text in their biographies which directly signal the topic *Minimal Art*, this demonstrates topical classifications which were achieved by our label spreading stage.

**Search Results.** The content creators are shown below the search box, where each panel shows public data from their Instagram profile including their *username*, *website* (in bold), biography and various stats about their profile. As a part of the statistics we include our calculation of engagement[1], which indicates how engaged the content creator is with their audience. Below the statistics, we present the most

[1]The average number of likes over last 9 posts divided by the number of followers.

recent 6 posts to provide a *visual* means to search.

**Infinite Scroll.** The results (content creators) are displayed in the form of an *infinite scroll*, specifically as the user scrolls through content, more content is loaded automatically. In comparison to pagination, infinite scrolling provides a more responsive experience, in that the user can efficiently browse an ocean of information without having to wait for the page to pre-load.

### 8.3.1 Search Features

Our search engine provides an array of features that allow users to navigate the database efficiently. Below we outline the features in our implementation:

**Topic Search.** A user is able to find content creators by selecting one or more topics from our taxonomy for which the content creators topics should match.

**Bio Search.** This feature enables the user to perform a full text search on biographies, querying for *phrases* which (i) must occur, (ii) must not occur or (iii) optionally occur in biographies. The optional occurrence simply ranks a search result higher if it matches, but does not enforce a match. Figure 8.2 demonstrates an example bio query in combination with a topic search with the intention to discover *Ceramic Shops*. Note that this figure exemplifies our *partial matching* feature which can be enabled or disabled below the sliders. This allows a user to find content creators who have a biography including the regular expression **shop**. One last note is that we allow users to perform searches for *phrases*, which can be seen as a synonym for *ngram* in this context. These allow users to find groups of words that occur next to each-other, for instance if a user wanted to search for a ceramic shop in London one could search bios for the phrases *ceramic shop* and *London*. This query would return results including those bios with the two phrases occurring at any position, but ensuring both *ceramic* and *shop* occurred one after the other in the correct order. Our search results highlight the results in the biographies in order to assist the user in finding relevant results. We implement a fast full text search by indexing all content creators streaming from our data collector into our Elasticsearch cluster (Section 6.1.3).

**Graph Search.** Here we provide an initial implementation of a graph search, whereby a user can find the content creators one content creator follows or is followed by. We believe it would be exceptionally useful for users when aiming to explore circles of content creators, providing use cases for both users and businesses. We query our Neo4j (Section 6.1.2) causal cluster via our REST API (Section 7.2.2) to provide low latency graph queries to the user.

**Filtering.** When searching for content creators, a user, or more likely a business may want to find content creators of a specific *engagement* or *follower size* which can be achieved through various sliders in our front-end.

**Location.** In this application we provide an initial implementation of a feature we plan to develop in the future work: *location search*. At this stage this application

allows a user to find content creators by location through matching the *most popular location tagged in all the content creators posts*. This is particularly useful for businesses and content creators where they would like to collaborate with content creators who are frequently in the same area as them.

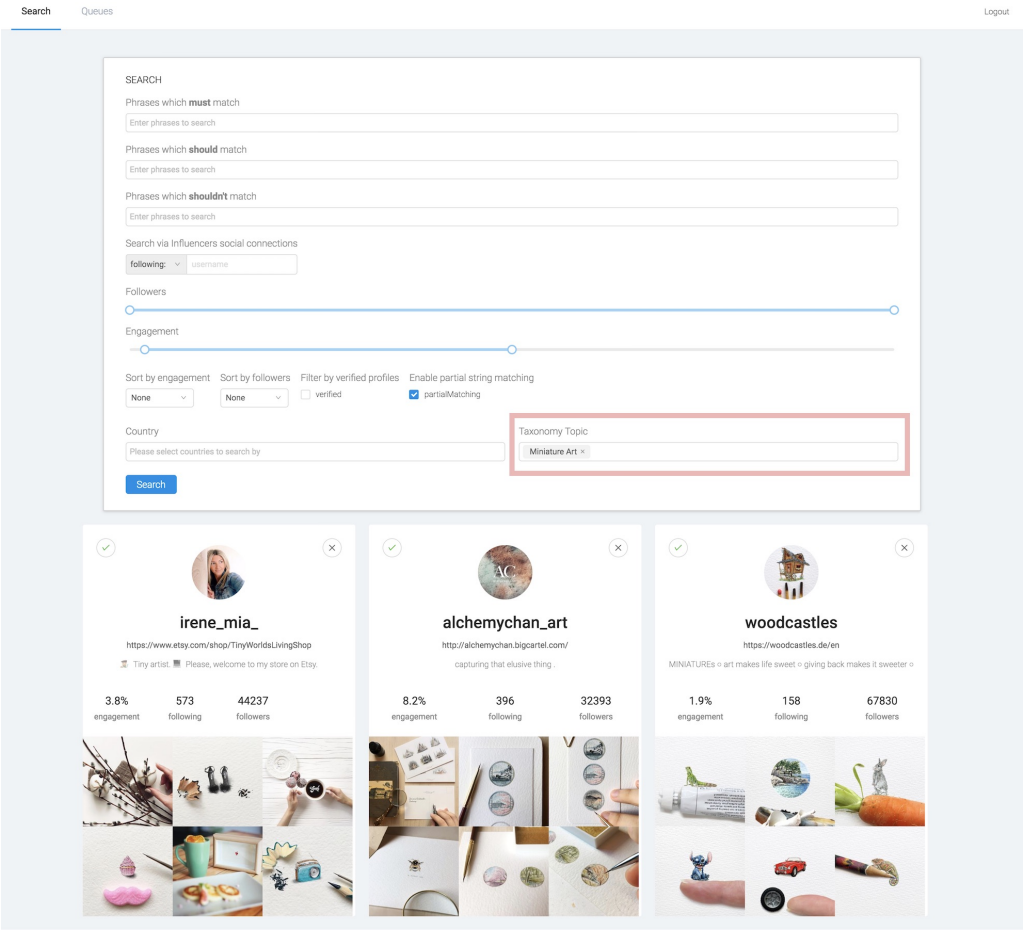



Figure 8.1: A screenshot demonstrating a search for *Miniature Artists* in our content creator search engine. The query is entered in the red box and the results are shown in the feed below the search box.

### 8.3.2 Experimental Evaluation

In order to perform effective experimental evaluation we aimed to design evaluation methods that removed as many human biases as possible. We initially considered an approach which was proposed in past research to evaluate *topic inference/classification*, whereby a human reviewer would select the number of relevant topics for an instance (e.g. document or content creator). This evaluation method would suffer from the same cognitive overload as discussed previously where a reviewer would have to enumerate an extremely large taxonomy to determine which to determine which topics are relevant. Instead, we aimed to utilise *corrective labelling* inspired by Yang et al. [88], whereby a human user determines whether a content creator is correctly or incorrectly labelled. We began this process by building an independent web application to gather this feedback by presenting users with a list randomly sampled (content creator, topic) pairs, however we decided that this approach would

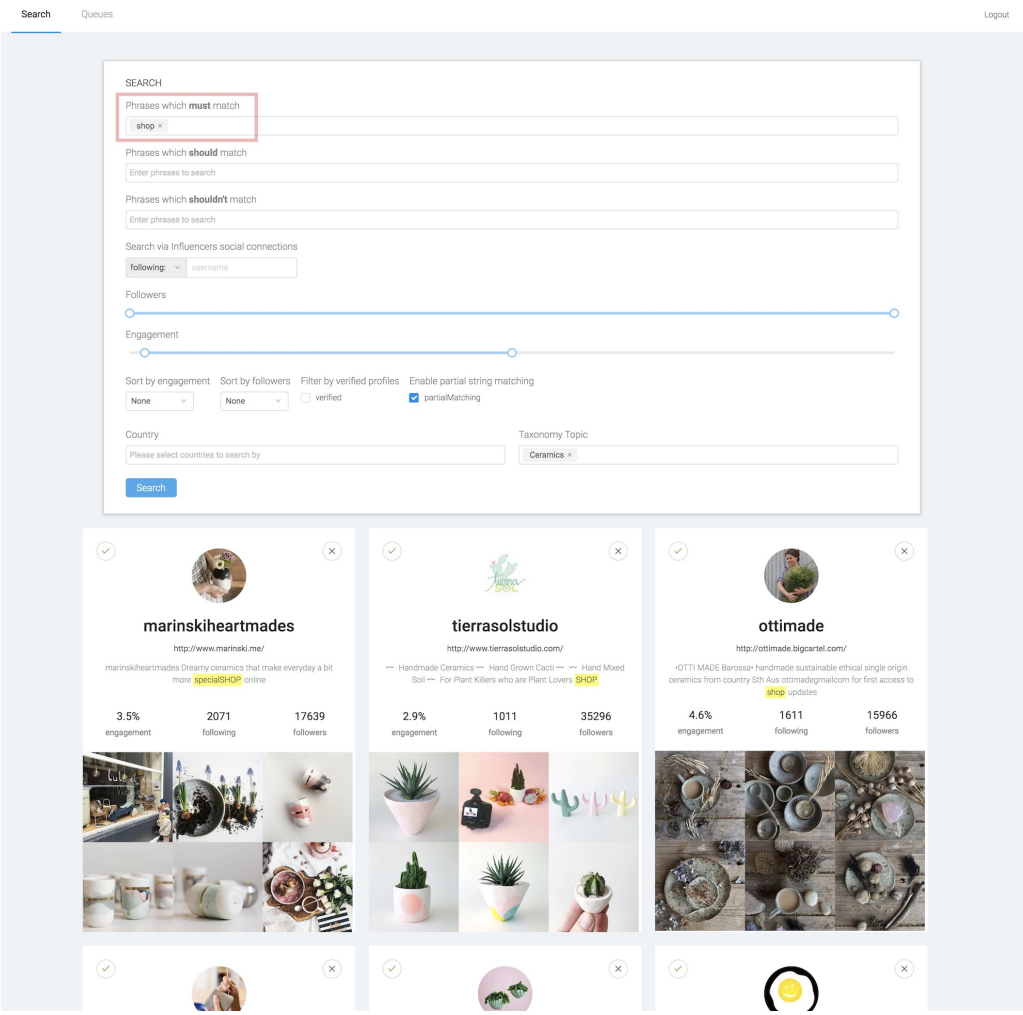


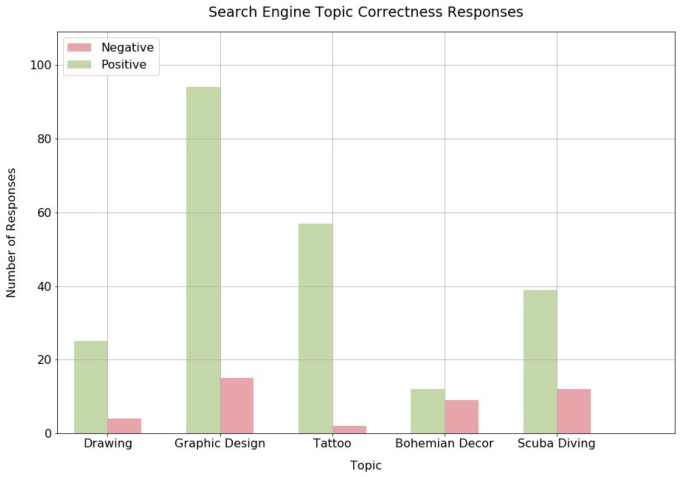

Figure 8.2: A screenshot demonstrating a search for *Ceramic Shops* by utilising full text biography search.

be excessive, particularly given the time restrictions.

**Integrated User Feedback.** Our instantiation of *corrective labelling* included integrating feedback mechanisms into our search engine, allowing users to provide feedback continuously through the lifetime of the application. Specifically, a user can "reject" a content creator if it is returned incorrectly when searching for a specific topic, additionally one can provide positive feedback where they believe a content creator is particularly well categorised. For instance, if a user was searching for *Painters* but a *Yoga* content creator (Figure 8.3b) were returned in the search results, a user can click the *reject* button shown in the top right corner of the result panel in Figure 8.3b. If a user believed that the content creators was effectively classified, they might provide *positive* feedback by clicking the *tick* in the top left corner of the same figure. This data is collected to provide feedback for what topics our architecture are classified well.

**Corrective Methods.** This allows us to determine the degenerate topics and debug issues, for example our dictionary classifier had a degenerate mapping between `Fish` and `Sushi` which classified all fishermen as *Sushi*. The large number of rejections in

fishermen lead to an investigation to remove this incorrect mapping. In future work we aim to incorporate a more automatic approach to reacting to user feedback, where the system could *actively learn* from its mistakes by retraining based on batched user feedback. We present an example of the feedback we acquired from users whilst interacting with the application in Figure 8.3a. Here we can observe that *Bohemian Decor* is often incorrectly classified, whereas *Tattoo* has high positive to negative ratio. In future work we aim to explore more rigorous evaluation methods for our user application through collecting data over a large time span, where we're able to collect feedback across all topics. It is important to note also this form of feedback is only useful for a single label classification problem, for multi-label approaches we would ask the user to provide feedback on which topic the content creator is falsely classified under. Through this feedback we established that our architecture performs exceptionally well over most topics we performed evaluation over, but far from finished. We discuss how we are continuing to extend this work in future work (Section 9.1), for specific directions we aim to explore for the specific components, we direct the reader to the future work sections of the relevant chapter.




(a) Results of the feedback collected from users on our search engine. For each topic we show the number of *positive* and *negative* feedback responses. A positive response indicates that a content creator is incorrectly classified, and negative indicates that the content creator is not indicative of the topic searched for.


(b) A content creator panel including the feedback buttons for users to provide *positive* and *negative* classification of a content creator.

# 9 Conclusions

In this work we described a complete architecture for classifying Instagram accounts into their topic of content, from the definition of user-facing products to the implementation details. This architecture and its accompanying products are deployed in production and are used to achieve various business goals for Filli Studios, a social media marketing agency. The individual components of the architecture are evaluated in terms of precision and recall, showing that, with a high-coverage taxonomy, it performs exceptionally well. Additionally we evaluate the architectures overall performance through its user-facing product in the the form of user feedback. To the best of our knowledge, this work is the first attempt in classifying the niche topics of content creators on Instagram at scale.

We hope the architecture designed and implemented in this work provides insights in how techniques from various fields can be applied to social media data science. These techniques include spanning semantic web, community detection, supervised and semi-supervised learning, among others. We found by combing insights from many fields enabled us to effectively solve our challenge of discovering and classifying topical content creators on Instagram.

Our future work is guided by the goal of improving precision and recall of our components and the coverage of our taxonomy. The individual improvements to each component can be found in their accompanying chapters.

## 9.1 Holistic Future Work

For the purpose of clarity and relevance, we provide a per-component detailed evaluation at the end of each chapter. In this section we provide a holistic approach, whereby we consider only future work which is related to all the components as a whole. A common observation we made during our evaluation procedures is that the more niche the topic, the lower the precision we're able to achieve. This we are confident is mostly a consequence of the following observations:

- **Generality of Topical Signals.** The limited descriptive power of biographies in providing topical signals of niche topics (e.g. Scuba Diving) prevented our dictionary classifier to extract the niche topics.
- **Noise from Entity Linkers.** It was often the case that entity linkers used in this work failed to link mentions to conepts in a knowledge base. One

approach we aim to explore to mitigate this issue, and hence improve our topical coverage is by extending our dictionary with phrases. For instance the phrase `yoga teacher` would map to the topics `Yoga` and *Fitness Instructor*.

### 9.1.1 Reliance on High Coverage Taxonomy

In Chapter 4 we defined a component of our architecture which is able to identify seed content creators for each topic in our taxonomy. The labels of the seed content creators are then spread to the rest of the content creator graph using the algorithms developed in Chapter 5. Through extending state-of-the-art methods, each component individually has been shown both empirically and quantitatively to perform extremely well for its designed purpose (i.e. *high-precision* and *high-recall*). It is important to note that for the topic inference pipeline to infer the topics of seeds with high-precision, the topics must be defined in our taxonomy, i.e. if a content creator has the topic *Van Life* which is not in the taxonomy, they will not be labelled (or be labelled a general category, e.g. *Travel*). Next, the label spreading algorithm can only spread labels which are collected by the topic inference pipeline, and by transitivity, represented by our taxonomy. Finally, users of the search engine application will not be capable of indexing unrepresented topics and the recommendations system may lack information to make "good" recommendations. In conclusion, *later components rely heavily on previous ones*, in particular, all are grounded upon the topical coverage of our taxonomy. Due to time restrictions we were unable to discover all possible possible topics and hence provide limitations in later components, preventing maximal performance. Additionally, missing topics may result in some false positives which can be observed through the following example:

> Imagine a content creator **C** with topic *Poetry* who follows 35 *Poets* and 5 *Music Artists*, where *Poetry* was not represented in our taxonomy, but *Music Artist* was. From this information, the topic inference pipeline would gather seeds for *Music Artist* (but not unknown topic *Poetry*) and the label spreading algorithm will infer that **C** is a *Music Artist*.

To eliminate these false positives we highlight the importance of the future work involved in development of the taxonomy. In summary, we conclude that our evaluations provide strong evidence that our architecture will infer the topics of content creators with high precision and high recall, *given the information it has*. This motivates us continue to use our topical community detection algorithms to discover topics and build a taxonomy which covers as many topics as possible.

### 9.1.2 Topical Similarity

Figure 5.2 shows an excellent snapshot of the expressive power of this work: the ability to determine topical similarity by measuring the similarity between the vector representations of topical communities. This vector representation can be generated by using graph embedding approaches like Node2Vec [30] which converts a graph into low dimensional space whilst preserving graph properties and information. Interested readers are directed to an review on the field by Cai et al. [9]. One may use this information to discover unexpected relationships between topics, or build an API to provide a programmatic means of querying similarity between niche topics on Instagram.

# Bibliography


[1] Bruno Abrahao, Sucheta Soundarajan, John Hopcroft, and Robert Kleinberg. On the separability of structural classes of communities. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '12*, page 624, 2012. ISBN 9781450314626. doi: 10.1145/2339530.2339631.

[2] Jimit Bagadiya. 171 Amazing Social Media Statistics You Should Know in 2018. https://www.socialpilot.co/blog/social-media-statistics. Retrieved: 10-01-18.

[3] Parantapa Bhattacharya, Muhammad Bilal Zafar, Niloy Ganguly, Saptarshi Ghosh, and Krishna P. Gummadi. Inferring user interests in the Twitter social network. In *Proceedings of the 8th ACM Conference on Recommender systems - RecSys '14*, 2014. ISBN 9781450326681. doi: 10.1145/2645710.2645765.

[4] Steven Bird and Edward Loper. Nltk: The natural language toolkit. *CoRR*, cs.CL/0205028, 2002.

[5] Roi Blanco, Giuseppe Ottaviano, and Edgar Meij. Fast and Space-Efficient Entity Linking in Queries. *Journal of Statistical Mechanics: Theory and Experiment*, 2015. doi: 10.1145/2684822.2685317.

[6] Roi Blanco, Giuseppe Ottaviano, and Edgar Meij. Fast and space-efficient entity linking for queries. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, WSDM '15, pages 179–188, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3317-7. doi: 10.1145/2684822.2685317.

[7] David M Blei, Blei@cs Berkeley Edu, Andrew Y Ng, Ang@cs Stanford Edu, Michael I Jordan, and Jordan@cs Berkeley Edu. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[8] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory - COLT' 98*, pages 92–100, New York, New York, USA, 1998. ACM Press. ISBN 1581130570. doi: 10.1145/279943.279962.

[9] HongYun Cai, Vincent Wenchen Zheng, and Kevin Chen-Chuan Chang. A comprehensive survey of graph embedding: Problems, techniques and applications. *CoRR*, abs/1709.07604, 2017.

[10] Diego Ceccarelli, Claudio Lucchese, Salvatore Orlando, Raffaele Perego, and Salvatore Trani. Dexter 2.0 - an open source tool for semantically enriching data. In *International Semantic Web Conference*, 2014.

[11] Francois Chollet. Using pre-trained word embeddings in a Keras model. https://blog.keras.io/using-pre-trained-word-embeddings-in-a-keras-model.html. Retrieved: May 9 2018.

[12] Yagmur Gizem Cinar, Susana Zoghbi, and Marie Francine Moens. Inferring User Interests on Social Media from Text and Images. In *Proceedings - 15th IEEE International Conference on Data Mining Workshop, ICDMW 2015*, 2016. ISBN 9781467384926. doi: 10.1109/ICDMW.2015.208.

[13] Aaron Clauset, M E J Newman, and Cristopher Moore. Finding community structure in very large networks. 70:066111, 01 2005.

[14] Joachim Daiber, Max Jakob, Chris Hokamp, and Pablo N. Mendes. Improving efficiency and accuracy in multilingual entity extraction. In *I-SEMANTICS*, 2013.

[15] Joachim Daiber, Max Jakob, Chris Hokamp, and Pablo N. Mendes. Improving efficiency and accuracy in multilingual entity extraction. In *Proceedings of the 9th International Conference on Semantic Systems (I-Semantics)*, 2013.

[16] Bhuwan Dhingra, Hanxiao Liu, Ruslan Salakhutdinov, and William W. Cohen. A comparative study of word embeddings for reading comprehension. *CoRR*, abs/1703.00993, 2017.

[17] Collins English Dictionary. lemmatise. (n.d.) Collins English Dictionary – Complete and Unabridged, 12th Edition 2014. https://www.thefreedictionary.com/lemmatise. Retrieved: May 8 2018.

[18] David. Easley and Jon. Kleinberg. *Networks, crowds, and markets : reasoning about a highly connected world*. Cambridge University Press, 2010. ISBN 9780521195331.

[19] Chantat Eksombatchai, Pranav Jindal, Jerry Zitao Liu, Yuchen Liu, Rahul Sharma, Charles Sugnet, Mark Ulrich, and Jure Leskovec. Pixie: A system for recommending 3+ billion items to 200+ million users in real-time. In *WWW*, 2018.

[20] Facebook. The New Universal Language - Facebook Insights. https://www.facebook.com/iq/articles/the-new-universal-language. Retrieved: May 4 2018.

[21] Emilio Ferrara, Pasquale De Meo, Giacomo Fiumara, and Alessandro Provetti. The role of strong and weak ties in facebook: a community structure perspective. *CoRR*, abs/1203.0535, 2012.

[22] Emilio Ferrara, Roberto Interdonato, and Andrea Tagarelli. Online popularity and topical interests through the lens of instagram. In *HT*, 2014.

[23] Thomas M. J. Fruchterman and Edward M. Reingold. Graph drawing by force-directed placement. *Softw., Pract. Exper.*, 21:1129–1164, 1991.

[24] Ullas Gargi, Wenjun Lu, Vahab S. Mirrokni, and Sangho Yoon. Large-scale community detection on youtube for topic discovery and exploration. In *ICWSM*, 2011.

[25] Saptarshi Ghosh, Naveen Kumar Sharma, Fabrício Benevenuto, Niloy Ganguly, and Krishna P. Gummadi. Cognos: crowdsourcing search for topic experts in microblogs. In *SIGIR*, 2012.

[26] Eva Gibaja and Sebast AN Ventura. 52 A Tutorial on Multilabel Learning. *ACM Comput. Surv. Article*, 47(52), 2015. doi: 10.1145/2716262.

[27] M Girvan and M E J Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences of the United States of America*, 99(12):7821–6, 6 2002. ISSN 0027-8424. doi: 10.1073/pnas.122653799.

[28] Erving Goffman. *The presentation of self in everyday life*. Penguin, 1990. ISBN 0140135715.

[29] Yoav Goldberg. Neural network methods for natural language processing. In *Synthesis Lectures on Human Language Technologies*, 2017.

[30] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. *KDD : proceedings. International Conference on Knowledge Discovery Data Mining*, 2016:855–864, 2016.

[31] Stephen Guo, Ming-Wei Chang, and Emre Kiciman. To link or not to link? a study on end-to-end tweet entity linking. In *HLT-NAACL*, 2013.

[32] Michael Hamann, Eike Röhrs, and Dorothea Wagner. Local Community Detection Based on Small Cliques. *Algorithms*, 10(3):90, 8 2017. ISSN 1999-4893. doi: 10.3390/a10030090.

[33] Aurélie Herbelot and Marco Baroni. High-risk learning: acquiring new word vectors from tiny data. In *EMNLP*, 2017.

[34] Jianbin Huang, Heli Sun, Jiawei Han, and Boqin Feng. Density-based shrinkage for revealing hierarchical and overlapping community structure in networks. *Physica A*, 390:2160–2171, 2011. doi: 10.1016/j.physa.2010.10.040.

[35] Jianbin Huang, Heli Sun, Yaguang Liu, Qinbao Song, and Tim Weninger. Towards online multiresolution community detection in large-scale networks. In *PloS one*, 2011.

[36] Rachsuda Jiamthapthaksin and Than Htike Aung. User preferences profiling based on user behaviors on Facebook page categories. In *2017 9th International Conference on Knowledge and Smart Technology: Crunching Information of Everything, KST 2017*, 2017. ISBN 9781467390774. doi: 10.1109/KST.2017.7886077.

[37] Systrom Kevin. Strengthening Our Commitment to Safety and Kindness for 800 Million. https://instagram-press.com/blog/2017/09/26/strengthening-our-commitment-to-safety-and-kindness-for-800-million. Retrieved: 10-01-18.

[38] Sheila Kinsella, Mengjiao Wang, John G. Breslin, and Conor Hayes. Improving categorisation in social media using hyperlinks to structured data sources. In *ESWC*, 2011.

[39] Jey Han Lau and Timothy Baldwin. An empirical evaluation of doc2vec with practical insights into document embedding generation. In *Rep4NLP@ACL*, 2016.

[40] Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents. In *ICML*, 2014.

[41] Roy Ka-Wei Lee, Tuan-Anh Hoang, and Ee-Peng Lim. On analyzing user topic-specific platform preferences across multiple social media sites. In *WWW*, 2017.

[42] Jure Leskovec and Christos Faloutsos. Sampling from large graphs. In *KDD*, 2006.

[43] Xiao Ling, Sameer Singh, and Daniel S. Weld. Design challenges for entity linking. *TACL*, 3:315–328, 2015.

[44] Chunliang Lu and Wai Lam. User modeling and tweets recommendation based on wikipedia concept graph . 2012.

[45] Miller McPherson, Lynn Smith-Lovin, and James M Cook. Birds of a Feather: Homophily in Social Networks. *Annual Review of Sociology*, 2001. ISSN 0360-0572. doi: 10.1146/annurev.soc.27.1.415.

[46] Rishabh Mehrotra, Scott Sanner, Wray L. Buntine, and Lexing Xie. Improving lda topic models for microblogs via tweet pooling and automatic labeling. In *SIGIR*, 2013.

[47] Matthew Michelson and Sofus A. Macskassy. Discovering users' topics of interest on twitter: a first look. In *AND*, 2010.

[48] Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[49] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality, 2013.

[50] George A. Miller and George A. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41, 11 1995. ISSN 00010782. doi: 10.1145/219717.219748.

[51] M. E. J. Newman and Martina S Girvan. Finding and evaluating community structure in networks. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 69 2 Pt 2:026113, 2004.

[52] Aastha Nigam, Salvador Aguinaga, and Nitesh V. Chawla. Connecting the dots to infer followers' topical interest on Twitter. In *IEEE/ACM BESC 2016 - Proceedings of 2016 International Conference on Behavioral, Economic, Socio - Cultural Computing*, 2017. ISBN 9781509061648. doi: 10.1109/BESC.2016.7804498.

[53] Fabrizio Orlandi, John Breslin, and Alexandre Passant. Aggregated, Interoperable and Multi-Domain User Profiles for the Social Web *.

[54] Aditya Pal, Amaç Herda, Sourav Chatterji, Sumit Taank, and Deepayan Chakrabarti. Discovery of Topical Authorities in Instagram. doi: 10.1145/2872427.2883078.

[55] Paula Peña, Rafael Del Hoyo, Jorge Vea-Murguía, Carlos González, and Sergio Mayo. Collective knowledge ontology user profiling for twitter: Automatic user profiling. In *Proceedings - 2013 IEEE/WIC/ACM International Conference on Web Intelligence, WI 2013*, 2013. ISBN 9781479929023. doi: 10.1109/WI-IAT.2013.62.

[56] Matthew E. Peters and Dan Lecocq. Content extraction using diverse feature sets. In *Proceedings of the 22nd International Conference on World Wide Web - WWW '13 Companion*, pages 89–90, New York, New York, USA, 2013. ACM Press. ISBN 9781450320382. doi: 10.1145/2487788.2487828.

[57] Guangyuan Piao and John G Breslin. Inferring User Interests for Passive Users on Twitter by Leveraging Followee Biographies. doi: 10.1007/978-3-319-56608-5.

[58] Guangyuan Piao and John G. Breslin. Exploring Dynamics and Semantics of User Interests for User Modeling on Twitter for Link Recommendations. In *Proceedings of the 12th International Conference on Semantic Systems - SEMANTiCS 2016*, 2016. ISBN 9781450347525. doi: 10.1145/2993318.2993332.

[59] Guangyuan Piao and John G. Breslin. User Modeling on Twitter with WordNet Synsets and DBpedia Concepts for Personalized Recommendations. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management - CIKM '16*, 2016. ISBN 9781450340731. doi: 10.1145/2983323.2983908.

[60] Guangyuan Piao and John G. Breslin. Analyzing aggregated semantics-enabled user modeling on google+ and twitter for personalized link recommendations. In *UMAP*, 2016.

[61] Yuval Pinter, Robert Guthrie, and Jacob Eisenstein. Mimicking word embeddings using subword rnns. In *EMNLP*, 2017.

[62] Xiao Pu, Mohamed Amine Chatti, Hendrik Thüs, and Ulrik Schroeder. Wikilda: A mixed-method approach for effective interest mining on twitter data. In *CSEDU*, 2016.

[63] Daniele Quercia, Harry Askham, and Jon A Crowcroft. Tweetlda: supervised topic classification and link prediction in twitter. In *WebSci*, 2012.

[64] Filippo Radicchi, Claudio Castellano, Federico Cecconi, Vittorio Loreto, and Domenico Parisi. Defining and identifying communities in networks. *Proceedings of the National Academy of Sciences of the United States of America*, 101 9:2658–63, 2004.

[65] Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 76 3 Pt 2:036106, 2007.

[66] Vikas C. Raykar, Shipeng Yu, Linda H. Zhao, Anna K. Jerebko, Charles Florin, Gerardo Hermosillo, Luca Bogoni, and Linda Moy. Supervised learning from multiple experts: whom to trust when everyone lies a bit. In *ICML*, 2009.

[67] Redis. What is Redis. https://redis.io/. Retrieved: May 21 2018.

[68] Pablo Ruiz and Thierry Poibeau. Combining open source annotators for entity linking through weighted voting. In **SEM@NAACL-HLT*, 2015.

[69] Indira Sen, Anupama Aggarwal, Shiven Mian, Siddharth Singh, Ponnurangam Kumaraguru, and Anwitaman Datta. Worth its weight in likes: Towards detecting fake likes on instagram. In *WebSci*, 2018.

[70] Wei Shen, Jianyong Wang, Ping Luo, and Min Wang. Linking Named Entities in Tweets with Knowledge Base via User Interest Modeling.

[71] Ajit Pal Singh and Arun C. Surendran. Recommendations using absorbing random walks. 2007.

[72] Dandan Song, Fei Sun, and Lejian Liao. A hybrid approach for content extraction with text density and visual importance of dom nodes. *Knowledge and Information Systems*, 42:75–96, 2013.

[73] Nemanja Spasojevic, Jinyun Yan, Adithya Rao, and Prantik Bhattacharyya. Lasta: large scale topic assignment on multiple social networks. In *KDD*, 2014.

[74] Mark Steyvers, Padhraic Smyth, Michal Rosen-Zvi, and Thomas Griffiths. Probabilistic Author-Topic Models for Information Discovery.

[75] Ana-Andreea Stoica, Christopher J. Riederer, and Augustin Chaintreau. Algorithmic glass ceiling in social networks: The effects of social recommendations on network diversity. In *WWW*, 2018.

[76] Yansen Su, Bangju Wang, and Xingyi Zhang. A seed-expanding method based on random walks for community detection in networks with ambiguous community structures. *Scientific Reports*, 7:41830, 2 2017. ISSN 2045-2322. doi: 10.1038/srep41830.

[77] Robert Tarjan. Depth-First Search and Linear Graph Algorithms. *SIAM Journal on Computing*, 1(2):146–160, 6 1972. ISSN 0097-5397. doi: 10.1137/0201010.

[78] Financial Times. User-Generated Content (UGC) Definition from Financial Times Lexicon. http://lexicon.ft.com/Term?term=user_generated-content-(UGC).

[79] Princeton University. wngloss - A glossary of terms used in WordNet system. https://wordnet.princeton.edu/documentation/wngloss7wn. Retrieved: May 16 2018.

[80] Xiaodong Wang and Hongkui Tu. Mining users' interest graph in social networks with topic based tag propagation. 2013:282–285, 01 2013.

[81] Joyce Jiyoung Whang, David F. Gleich, and Inderjit S. Dhillon. Overlapping community detection using seed set expansion. In *CIKM*, 2013.

[82] Scott A White and Padhraic Smyth. A spectral clustering approach to finding communities in graphs . 2005.

[83] Sanjaya Wijeratne, Lakshika Balasuriya, Amit P. Sheth, and Derek Doran. Emojinet: An open service and api for emoji sense discovery. In *ICWSM*, 2017.

[84] WordNet. What is WordNet? https://wordnet.princeton.edu/. Retrieved: May 3 2018.

[85] Wayne Xin ZHAO, Jing Jiang, Jianshu Weng, Jing He, Ee Peng LIM, Citation Zhao, Jiang Jing, Wng Jianshu, He Jing, Lim Ee-Peng, Yan Hongfei, Li Xiaoming, Author Wayne Xin ZHAO, Hongfei Yan, Wayne Xin Zhao, Ee-Peng Lim, and Xiaoming Li. Comparing Twitter and Traditional Media using Topic Models. 2011.

[86] Jiejun Xu and Tsai Ching Lu. Inferring user interests on tumblr. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2015. ISBN 9783319162676. doi: 10.1007/978-3-319-16268-3{_}59.

[87] Jaewon Yang and Jure Leskovec. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems*, 42(1):181–213, 1 2015. ISSN 0219-1377. doi: 10.1007/s10115-013-0693-z.

[88] Shuang-Hong Yang, Alek Kolcz, Andy Schlaikjer, and Pankaj Gupta. Large-scale high-precision topic modeling on twitter. In *KDD*, 2014.

[89] Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In *NIPS*, 2003.

[90] Ding Zhou, Eren Manavoglu, Jia Li, C. Lee Giles, and Hongyuan Zha. Probabilistic models for discovering e-communities. In *WWW*, 2006.

[91] Xujuan Zhou, Yue Xu, Yuefeng Li, Audun Josang, Clive Cox, X Zhou, Y Xu, Y Li, and A Josang. The state-of-the-art in personalized recommender systems for social networking. *Artif Intell Rev*, 37:119–132, 2012. doi: 10.1007/s10462-011-9222-1.

[92] Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, 2002.