Doctoral Dissertation

A study on the integration of machine learning and evidential reasoning in user preferences

Vo Duc Vinh

Supervisor: Professor Huynh Van Nam

Graduate School of Advanced Science and Technology
Japan Advanced Institute of Science and Technology
(Knowledge Science)

March 2022

Abstract

With the increase in use of social networks, users on these micro-blogging platforms usually share their thoughts and interests via short texts such as posts, status, or reactions. Capturing user preferences or interests from these kinds of data has attracted much attention. This process is known as the user profiling or user preferences. This problem aims at processing, inferring, and extracting a list of weighted keywords (or a semantic-based structure) that correctly represent the expertise or preferences of a specific user on these networking platforms. The problem has various potential applications in practice such as researcher finding in academic projects, item recommendation in e-commerce systems, or job offering in labour markets. However seeking an efficient solution is not trivial but a challenging task. Researchers working on this problem usually face following common challenges: (1) The *data sparsity* and *cold-start* issues existing in user texts; (2) user preferences dynamically *change* over time; (3) social networking users usually create lots of *short documents*. The consecutive documents are often not very closely related to each other. This causes difficulties for inferring the desired profile; (3) data may come in different formats (e.g., images, texts, or reactions) from multiple sources (e.g., one user may simultaneously have multiple accounts across networking platforms).

This research is motivated by three major factors. The first factor is reasoning ability in evidence theory (also preferred to as Demspter-Shafer theory). This theory is theoretically well-studied to become a full-fledged theory of uncertainty. It has been widely applied to various topics, including machine learning problems (e.g., classification and clustering), problems involving uncertainty (e.g., database management with uncertainty), and multiple-attribute decision making. The second factor is that advancements in machine learning and deep learning have shown significant remarkable achievements in both academic and practice. Machine learning practitioners proposed many robust, data-driven models that can learn from input observations to make accurate predictions, find hidden patterns, or even create new instances that are very similar to the input data. The third factor is the increasing of short texts shared by users on social networking platforms. Additionally, a number of open-sourced libraries make such kind of data obtainable and feasible to be processed.

Taking the aforementioned challenges and motivations into consideration, this research proposed two novel frameworks for *capturing user profiles using short texts under both static and dynamic scenarios*. The first framework is

designed for inferring *static* profiles, which is based on evidence theory and k-means clustering. The second framework is designed for inferring *dynamic* profiles, which is based on deep generative networks and evidence theory. In both proposed frameworks, advanced machine learning techniques, such as k-means clustering and deep generative neural networks, are used for concept learning from user short texts. The learned concepts form the so-called *frame of discernment* in Dempster-Shafer theory for reasoning process. These concepts are quantitatively transformed into the so-called *mass function* in the evident theory by *maximum a posterior estimation*. The derived mass functions are then considered as pieces of evidence at the reasoning phase. Finally, Dempster's rule and the so-called *pignistic probability distribution* are used for information fusion and profile extraction. The experiments on short text data sets verified that the proposed methods outperform baseline models on many evaluation metrics. Additionally, we also propose an approach for visualizing the fluctuation of user preferences on various topics over time by using the output of the proposed frameworks. This visualization may reveal significant insights that are useful for many practical applications.

Keywords: User profiling, user preferences, Dempster-Shafer theory, user profile visualization, deep generative networks.

Acknowledgments

First of all, I would like to express my sincere thanks to Professor Huynh Van-Nam for supervising me in the doctoral program. The knowledge and experience that Professor Huynh conveyed are precious for my research. Without his support, this research could not be completed. Some teachers help build characters. Some teachers inspire a love of learning. I would like to thank you Professor Huynh for being the teacher that does both.

Next, I would like to thank the Japanese Ministry of Education, Culture, Sports, Science and Technology (MEXT) for awarding me the Monbukagakusho scholarship. This award has provided a good financial support that helps me to pursue this research at JAIST. Additionally, I would like to thank Japan Advanced Institute of Science and Technology for offering me an actively professional, innovative research environment. The support from MEXT and JAIST holds a vital role in the completion of this research.

I also would like to thank Associate Professor Shirai Kiyoaki for supervising me on the minor research. Besides supervision on the minor research, Professor Shirai also collaborated with me on my publications. His support and suggestions are precious for improving the quality of the minor research as well as the publications.

Next, I would like to thank Professors in the evaluation committee, Professor Inuiguchi, Professor Hashimoto and Professor Dam, for their constructive comments and suggestions. These comments are helpful for improving the quality of my thesis.

Finally, I would like to thank my lab-mates at Huynh Lab and friends for their encouragement and support not only in my research but also in the daily life. Without the support from friends and lab-mates, I think that my life at JAIST would have become more challenging than it was.

List of Figures

List of Algorithms

List of Tables

Contents

Chapter 1

Introduction

1.1 User profiles and user profiling problem

With the dramatic increase in use of social networks, many users usually share their thoughts or emotions via these micro-blogging platforms. Therefore, a large amount of data associated with these platforms are created in a plentiful formats such as texts, photos, or reactions. This trend brings opportunities for organizations, especially e-commerce companies. If a company is able to *build a user model* for mining such kinds of data, then it could be able to *detect user preferences* and recommend *appropriate* products basing on the detected insights. These insights may reveal many practical applications such as *marketing campaigns* for the launch of new products, *job offering* on labour markets, or *user clustering* in customer management systems. However, this mining process is not a trivial, but a challenging task. Researchers working on this task usually face with questions such as how to *collect* and *normalize* user data, or how to *extract* and *combine* useful information from such kind of data for building the desired user model. That process is known as the **user profiling** or **user preferences** problem. And this research aims at finding an efficient approach for the problem. Figure 1.1 provides an overview picture of the user profiling process.

User Profiles. A user profile is a collection of information pieces associated with a specific person. These information pieces could be either personal identity (e.g., name, age, portrait picture, and locations) or individual characteristics (e.g., knowledge, expertise, and hobbies) or both. Figure 1.2 demonstrates an example of digital user profiles. There are two kinds of profiles, physical user profiles (e.g., resident cards, passports, or driving license) and digital user profiles (e.g., a virtual representation of individual

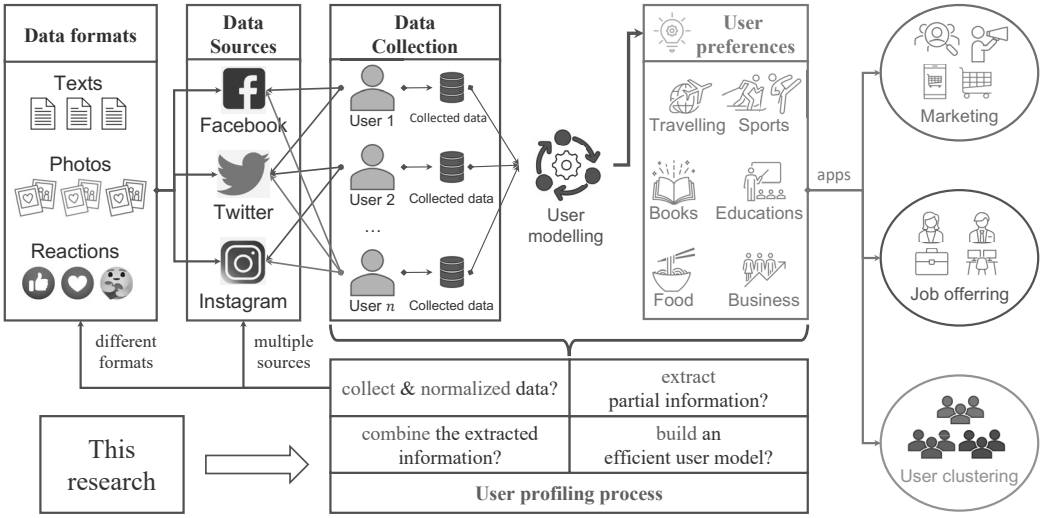


Figure 1.1: An overview of the user profiling problem.

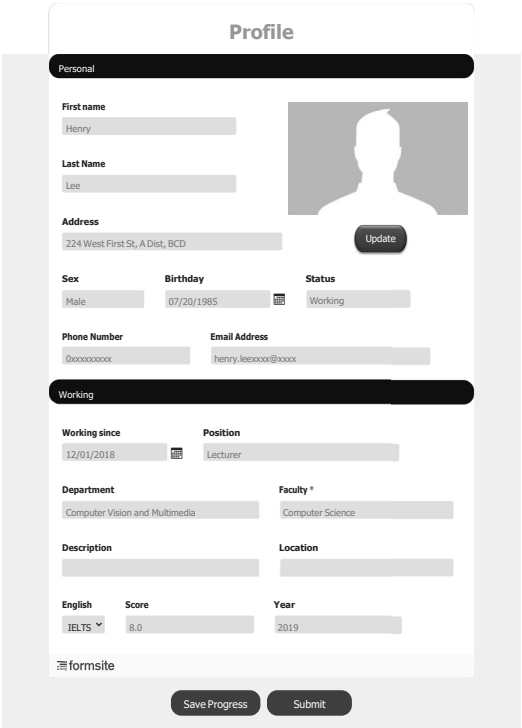


Figure 1.2: An example of a digital user profile.

data in computer systems associated with a specific user account). This research solely focuses on digital user profiles, more specifically profiles on social media. A social networking profile of one person is a record of skills

or preferences plus a network description of that person [3].

User Profiling Problem. The problem of user profiling aims at identifying a list of weighted keywords (or a semantic-based structure) for each account that represent the expertise or preferences of that user [3, 4, 5, 6].

An accurate representation of user profiles is crucial in various business contexts, especially item recommendation in e-commerce systems. The next section will introduce some applications of user profiles in practice.

1.2 Applications of user profiles

This part briefly introduces practical applications that motivates the user profiling problem, including researcher profiles, and item recommendation.

Researcher profiles. The first application is the construction of profiles for researchers on ArnetMining system[1][1]. One basic requirement in this system is to built a profile for each researcher. Each profile usually contains basic information (e.g., photo, affiliation, position, and contact), educational background (e.g., university graduated from and major), research interests, and key publications. Each profile could be built from the partial information extracted from multiple sources on web environments. For example, basic information and educational background may come from the researcher homepage or from a Web page that introduces him/her, information about publications may be integrated from online digital libraries (e.g., DBLP, IEEE, or ACM), and research interests could be deduced from a collected database. Figure 1.3 illustrates one instance from researcher profiles. The left-top part shows the researcher homepage, the left-bottom shows the DBLP/ACM page listing the publications. The right part of the figure display the ideal profile of that researcher. It contains research interests mined from the publication papers. If all researcher profiles are constructed correctly, we will have a large collection of well-structured database about researchers around the world. Then this database can benefit many practical tasks such as expert finding on a given topic or a research project.

Item recommendation. Users on e-commerce systems usually change their concerns on different product types over time. One crucial requirement in such systems is how to correctly capture the dynamics of these changes in order to recommend appropriate products for a given time interval. Inferring

[1] http://www.arnetminer.org

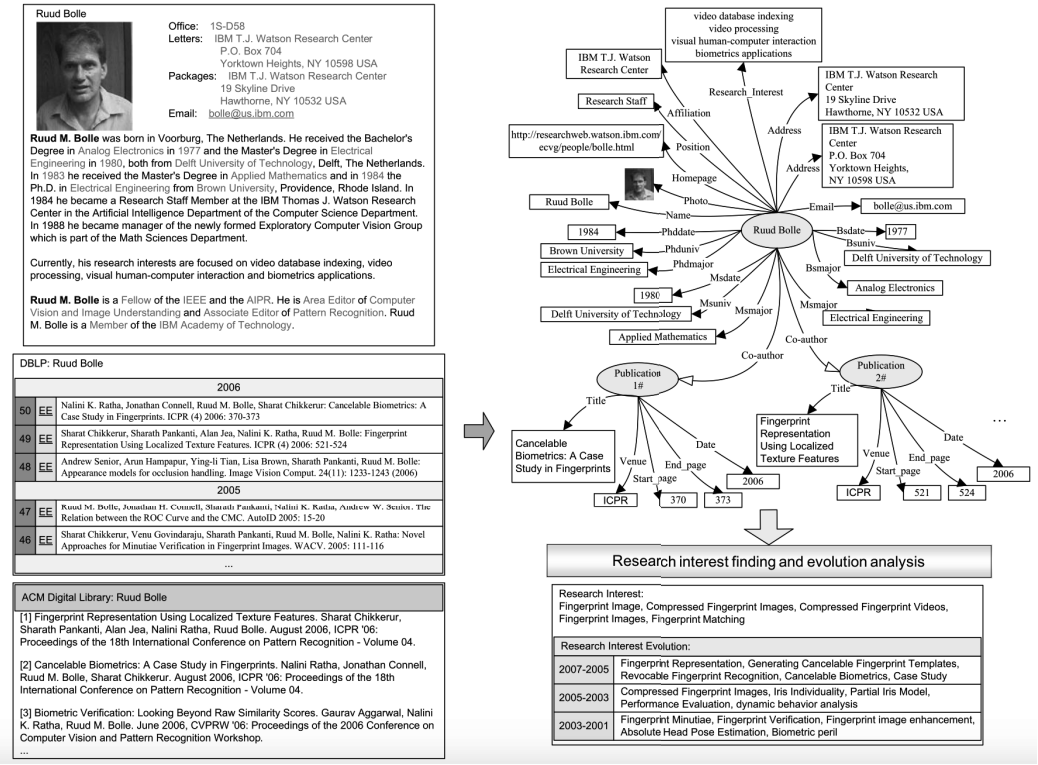


Figure 1.3: An instance from researcher profiles (source: [1]).

user concerns and visualization the results can benefit the recommendation process. Figure 1.4 visually depicts the "interest" level of two Twitter users on five given topics (Education, Economics, Science & Technology, Entertainments, and Politics) over time. Some insights could be drawn from this visualization: (1) the most concerned topic of the user 1 is Education while the most concerned topic of the user 2 is Entertainments; (2) the level of concern on Politics is on the upward trend for the user 1. These observations are useful for item recommendation. For example, we can suggest a discount coupon which can be used for online courses for the user 1, or suggest cinema tickets or sport equipment for the user 2.

1.3 Motivations of the research

Three major factors that motivate this research are advancements in machine learning and deep learning, reasoning ability in Dempster-Shafer theory of evidence, and the increase of obtainable short texts shared by users on social networking platforms.

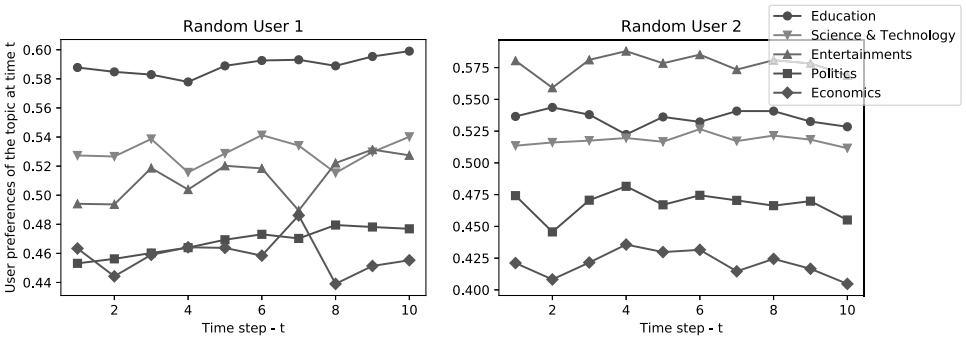


Figure 1.4: The changes of Twitter user preferences over time.

Advancements in machine learning and deep neural networks. In two recent decades, research on machine learning and deep learning has brought remarkable achievements in various machine learning problems, including supervised learning, semi-supervised learning, and unsupervised learning [7, 8, 9]. Practitioners have proposed many robust, data-driven models that can learn from input observations to make accurate predictions, find hidden patterns, or even create new instances that are very similar to the input data (e.g., generative models) [10, 11, 12]. Additionally, machine learning has been widely applied to real-world products that enhance both industrial processes and human life. The applications should be mentioned are image recognition (e.g., face recognition in smart home systems), speech recognition (e.g., Siri, Cortana, or Google Assistant), and text processing (e.g., spam filters, smart replies, or plagiarism checking).

Dempster-Shafer theory and its applications. Dempster-Shafer theory (DST) was first introduced by Dempster [13], then developed by Shafer [14]. It is a general approach for modeling and reasoning with uncertain, imprecise information. DST is understood as the extension of propositional logic and probabilistic reasoning. The theory is proven as a generalization of Bayesian inference [15]. DST is feasibly applied in practice by decomposing the available evidence into elementary pieces of evidence according to a certain perspective. Then these pieces are combined by an appropriate operator called Dempster's rule of combination. Thanks to this property, practitioners can apply the theory for solving problems where there may be a chance that different evidence may lead to different results. Additionally, this theory can be applied to a large-scaled problem [16]. These features makes DST applicable to a wide range problems involving uncertainty. Started from the work on statistical inference [13, 14, 15], DST is theoretically well-studied to become a full-fledged theory of uncertainty, and now commonly referred to as

Dempster-Shafer (DS) theory or evidence theory [17, 18]. It has been applied in many research topics, including knowledge-based systems [19], multiple-attribute decision making [20], association mining [21], collaborative filtering [22], database management [23], image processing [24, 25], pattern recognition [26], target tracking [27, 28, 29], information fusion [30], classification [31, 32], and clustering [33, 34].

The increase of obtainable short texts on networking platforms. With the increase in use of social networks in recent years, users on these platforms tend to express their feelings or share their personal characteristics via short texts. Capturing the dynamics of user interests in a correct manner using these short texts plays an essential role in various business contexts, especially in e-commerce systems. Besides, a variety of open-source libraries make it feasible to collect user textual data, and make the process of transforming these data into a fine, well-structured format possible. As a result, a collection of informative data sets becomes obtainable, and is ready to serve as input for building user models. One advantage of text-based user profiling problem is that the models can be built at a low cost, and be feasible to be deployed in practice.

Inspired by these factors, this research aims at incorporating the advancements of machine learning and Dempster-Shafer theory of evidence into proposing novel, robust models for the user profiling problem using short texts collected social networking platforms.

1.4 Research objectives and approaches

Although this research has many potential applications in practice, seeking an efficient solution is a challenging task. This is because of two main reasons. The first reason comes from the nature of short texts created by social networking users. The second reason arisen when the problem is considered under a *dynamic* context. These challenges are summarized in **Figure** 1.5.

In more details, researchers working on this problem usually face common challenges as follows:

- The *data sparsity* and *cold-start* issues existing in short texts. Data sparsity issue means that the amount of input data within a specific time interval is limited. This is the case when user corpus is divided into small batches according to timestamp for inferring the dynamics of user interests. Cold-start issue appears when the desired user model needs to capture/generate *fresh*, *appropriate* words/tokens that first appear *anywhere* in user corpus.

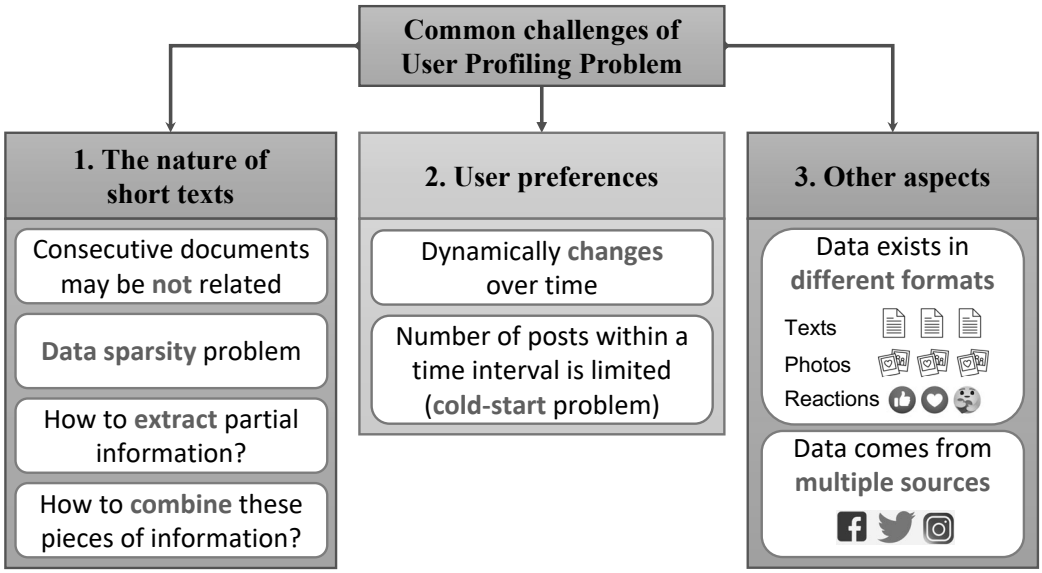


Figure 1.5: Common changes of the user profiling problem using short texts.

- User preferences dynamically *change* over time. This attribute requires a mechanism to *connect* the information extracted from the current batch of texts and the information extracted from the historical texts when building the model. This is necessary for capturing the fluctuations of user interest over time, which is valuable for decision making (e.g., the concerning level of the user 1 on Politics is on the upward trend over time as shown in Figure 1.4).

- Social network users create lots of short documents. This leads to some related issues such as short text mining and information selection. Short text mining is the ability to *extract* and *combine* useful information from these documents to identify user preferences. In many cases, practitioners need to mine the abstract concepts that are stored behind plain texts. Information selection is the ability to relate *relevant* information pieces from a pool of extracted information for building user models. This case occurs because consecutive documents are often not very closely related to each other in terms of content. As a result, the distribution of user concerned topics is nearly uniform or divergent. For example, a social networking user may share their thoughts about politics on the first post, introducing an charity event in the second post, and marketing a product in the third post. This case explains why previous approaches such as Topic Models (e.g., [35, 36, 35]) and Sequence Models (e.g., [37]) do not work well on such kind of data.

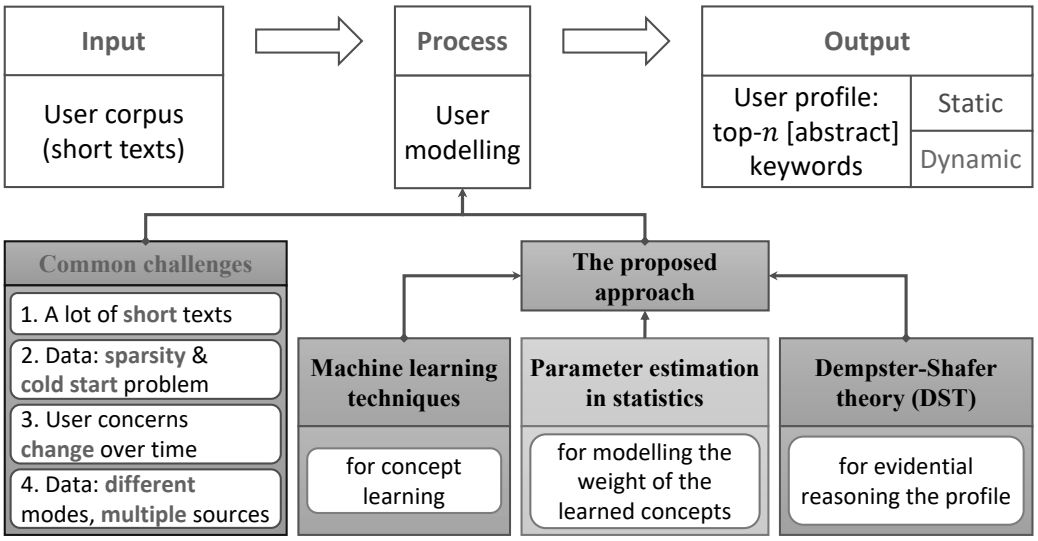


Figure 1.6: An integration of machine learning techniques and evidential reasoning for learning user preferences.

- Data may come from *multiple sources* (e.g., one user may simultaneously has multiple accounts on different social networking platforms). The data in each source could exist in *different formats* (e.g., texts, photos, and reactions). Seeking a robust model for coping with these attributes is still a research issue.

Taking these challenges into consideration, the main objective of this research is to propose novel approach(es) for *capturing profiles (or preferences) using short texts made by social networking users.* The problem is considered under two scenarios, the **static** context and the **dynamic** context. The problem is formally defined as follows:

Problem Formulation. Given a corpus of short texts made by social networking users (e.g., Twitter, Facebook, or Instagram), the objective of this problem is to dynamically extract a semantic profile that represents user preferences on various topics (e.g., education, science & technologies, entertainments, politics, and economics) at a specific time t. The results are represented by a list of keywords that are most *similar* to words in user vocabulary. Mathematically, the objective is to find a function f such that:

$$f^t : \langle \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{D}^t} \rangle \mapsto \boldsymbol{\mathcal{W}^t}$$

where $\boldsymbol{\mathcal{U}} = \{u_1, u_2, ..., u_m\}$ represents a set of users, with u_i being the i^{th} user in $\boldsymbol{\mathcal{U}}$ and $m = |\boldsymbol{\mathcal{U}}|$ is the total number of users; $\boldsymbol{\mathcal{D}^t} = \{D_1^t, D_2^t, ..., D_m^t\}$ is

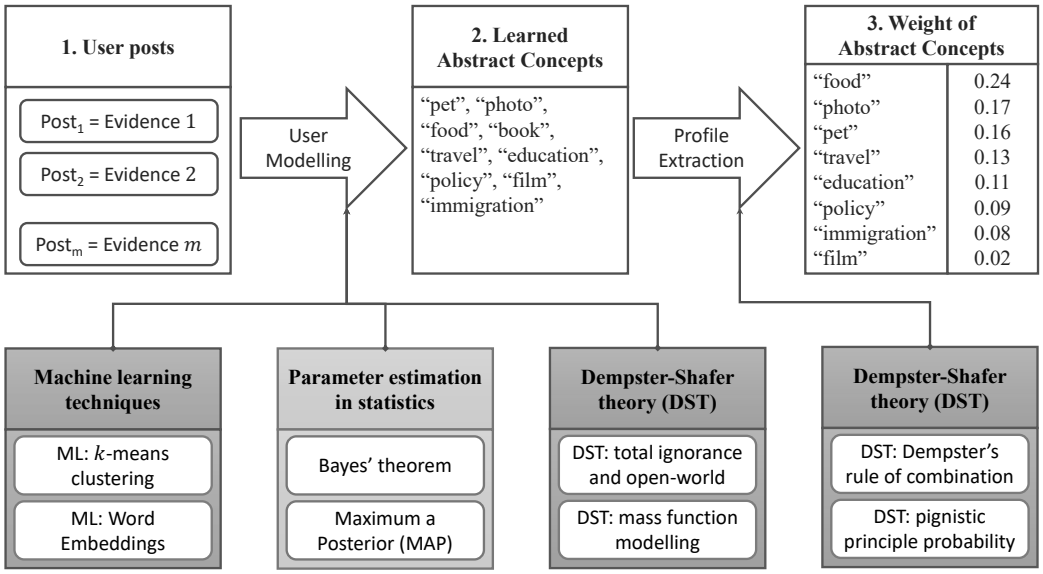


Figure 1.7: An intuitive view point of the proposed framework for learning static profiles.

the set of all users' corpora, each corpus D_i^t stores all posts (short documents) made by the corresponding user u_i within the time span t (in the case of static profile, the number of time spans is equal to 1); and $\boldsymbol{\mathcal{W}^t} = \{\boldsymbol{kw_1^t}, \boldsymbol{kw_2^t}, ..., \boldsymbol{kw_m^t}\}$ represents all users' preferences over time with $\boldsymbol{kw_i^t} = \{kw_{i,1}^t, kw_{i,2}^t, ..., kw_{i,n}^t\}$ being the *topic* that the i^{th} user concerns at time t, and the top-n keywords are obtained such that they are *most similar* to words from that user u_i's vocabulary.

To cope with the aforementioned challenges, the overall approach of this research is to find *an appropriate integration of machine learning techniques and evidential reasoning for inferring user profiles (or user preferences).* This approach is illustrated in **Figure** 1.6. Particularly, the research objective is divided into sub goals, each of which is accompanied by an appropriate approach described as follows:

- Obtaining a collection of data from social network users: we develop computer programs for automatically crawling textual data from social networks, processing, and storing the pre-processed data into a mini, unique data "warehouse" which enables the later phase to learn and extract information efficiently.

- Representation learning from pre-processed data for the reasoning process: we apply advanced machine learning techniques such k-means clustering and deep neural network models for learning hidden spaces.

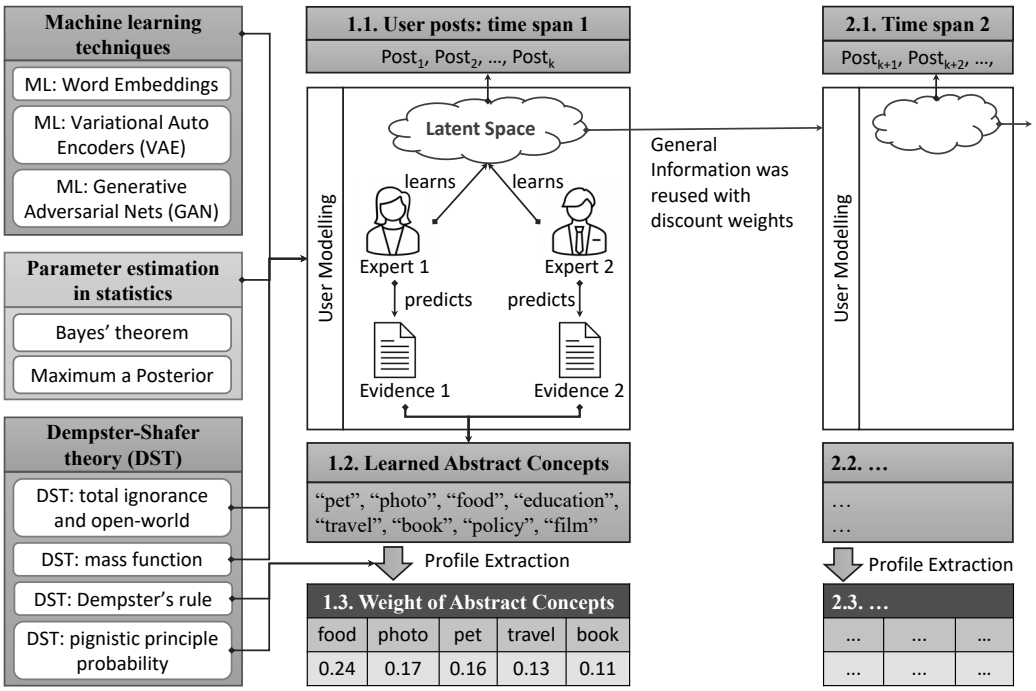


Figure 1.8: An intuitive view point of the proposed framework for learning dynamic profiles.

Then the learning results are represented at an appropriate abstraction level which is useful for the inferring task. Besides, we design models which are able to generate meaningful words that semantically represent user preferences based on the general information extracted from user texts within a specific time span.

- Reasoning and modeling user profiles from the extracted information: firstly, we find a mathematical approach to quantitatively transform information learning in previous tasks into a format that could be modeled by the so-called mass function in evidence theory. These masses are considered as pieces of evidence, which are used for reasoning purpose. Then, we apply Dempster's rule for combining partial information from multiple sources into a unified format that is useful in decision making when extracting the desired profile.

1.5 Key contributions of the research

This research results in two novel frameworks for the user profiling problem. The first framework is designed for inferring *static* user profiles, which is intuitively illustrated in **Figure** 1.7. This framework is based on evidence theory and k-means clustering. The second framework is designed for inferring user profiles in a dynamic context, which is intuitively depicted in **Figure** 1.8. This framework is based on deep generative networks and evident theory. Technical details of these two frameworks will be described in **Chapter 3** and **Chapter 4**, respectively. In summary, the key contributions of these two frameworks are briefly summarized as follows:

- Advanced machine learning techniques are utilized for concept learning from user short texts. Particularly,
  - In the first proposed framework, k-means clustering algorithm is employed to learn the abstract concepts that are stored inside user plain texts. These concepts are represented at multiple levels of abstraction, which are useful for the modeling process.
  - In the second proposed framework, two deep generative networks, the Variational AutoEncoder (VAE) and the Generative Adversarial Network (GAN), are used to separately learn the latent spaces of user texts. The trained networks then work independently to generate bunches for modelling user profiles.
  - The learned concepts form the so-called **frame of discernment** in Dempster-Shafer theory. Then these frames are used for modeling and extracting top-n keywords to form the user profile.
- We propose a mathematical approach which utilizes maximum a posterior estimation (MAP) in statistic to quantitatively transform the learned concepts into the so-called mass function in evidence theory. These masses are then viewed as pieces of evidence while inferring and determining the user profile.
- We propose an approach for visualizing the fluctuation of user preferences over time by using the output of the proposed frameworks (Figure 1.4). This visualization may reveal significant insights that are useful in many practical applications.
- We integrate two concepts in Dempster-Shafer theory, the *total ignorance* and the *open-world*, into the proposed models to capture fresh tokens that may first appear anywhere in user corpus (the cold-start

issue). This approach is useful when considering the dynamics of user preferences over time.

- Overall, we propose a new integrated approach in which advanced machine learning techniques are paired with Dempster-Shafer theory to tackle the aforementioned issues in user profile learning.

1.6 Organization of the research

This dissertation consists of five chapters. The connection between chapters is demonstrated in Figure 1.9. Also, main content of each chapter is described as follows:

Chapter 1 introduces an overview of the research. It includes the definition of user profiles and user profiling problem, practical applications, motivations, objectives, approaches, and organization of the research.

Chapter 2 provides background knowledge which is necessary for the research. These backgrounds include the introduction of basic concepts in evident theory and machine learning techniques (such as k-clustering and deep generative neural networks). Additionally, this chapter also highlights the influent works related to user profiling problem in literature.

Chapter 3 presents in details the proposed framework for the static profile. It first introduces the proposed method, then shows experimental results on short text data sets. Finally, the chapter discusses some related issues.

Chapter 4 describes in details the proposed framework for the dynamic profile. It first introduces the proposed method, then shows experimental results on short text data sets. This chapter also provides explanation on why the proposed framework works efficiently on short texts in comparison with baseline models.

Chapter 5 concludes the dissertation. It provides a summary on the research, discusses the related issues, lists limitations, and introduces future works.

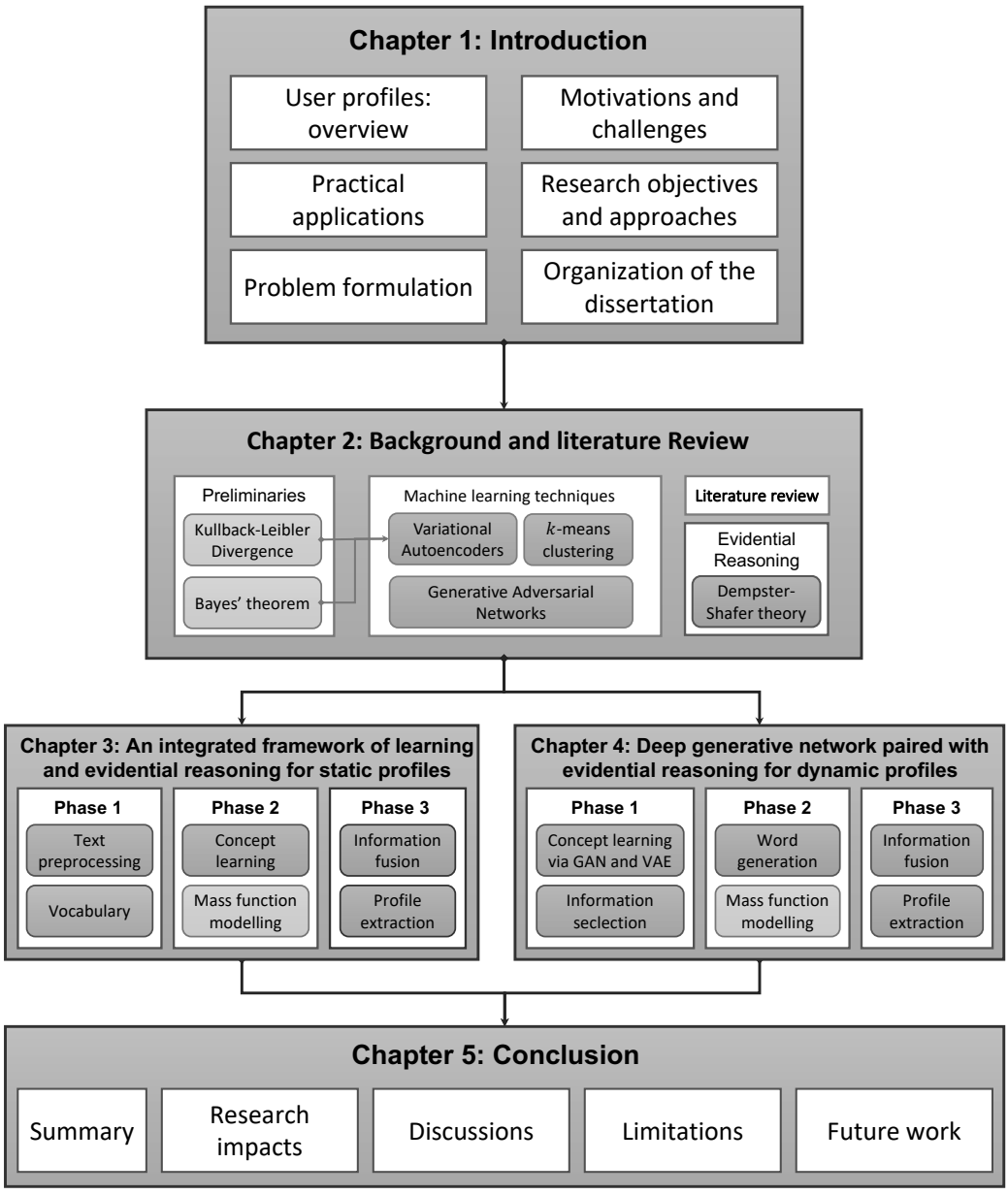


Figure 1.9: Organization of the dissertation.

Chapter 2

Background and Literature Review

2.1 Basics of evidence theory

Evidence theory was first introduced by Dempster [13], then developed by Shafer [14]. The theory provides a general approach for modeling and reasoning with uncertain, imprecise information pieces. The theory is also preferred as Dempster-Shafer theory (DST), theory of evidence, or theory of belief functions. It is proven as a more generalization of the Bayesian probability [15], and has been widely used in various fields such as knowledge-based systems [19], multiple-attribute decision making [20], image processing [24, 25], object monitoring [27, 28], pattern matching and recognition [26], [29], and information fusion [30]. In machine learning, DST is utilized to solve two typical problems, classification [31], [38] and clustering [33, 16]. This section will briefly introduce basic concepts of the theory which are used in this research.

2.1.1 Mass function

Let $\Omega = \{\omega_1, \omega_2, ..., \omega_N\}$ be a finite set Ω of N mutually exclusive, exhaustive hypotheses. These hypothesises are possible answers to a given question, and also preferred to as ***frame of discernment*** for that question [14]. A ***mass function*** m defined on Ω is a mapping:

$$m : 2^{\Omega} \mapsto [0, 1] \tag{2.1}$$

such that

$$m(\emptyset) = 0, \text{ and} \sum_{S \subseteq \Omega} m(S) = 1 \tag{2.2}$$

The mass function m is also preferred to as the ***basic probability assignment*** BPA) on Ω, and could be interpreted as below:

- The frame of discernment Ω contains possible answers to a given question. For example, in this research, the question could be "Which keyword in user vocabulary is the most important word that reflects user profile?".
- The quantity $m(S)$, where S is a subset of Ω, is the degree of belief exactly assigned to the proposition "the true answer is in S" and ***nothing more***. This means that the quantity $m(S)$ does not reveal any specific conclusion about the relationship between the actual answer and the subset of S).
- m describes one piece of ***evidence*** pertaining to that question.
- if $m(\emptyset) = 0$, then m is said to be ***normalized***. Otherwise, there may be some answers outside Ω (***open-world***).
- if $m(\Omega) > 0$, then this quantity quantitatively represents the concept ***total ignorance*** in DST.

.

The **focal set** of m, denoted as $\mathcal{F}_m$, is defined as the set of all subsets $A \subseteq \Omega$ satisfying $m(A) > 0$. In special cases, m degenerates to categorical, Bayesian, and consonant mass functions [15]:

- if the cardinality of A is equal to 1, for all A in $\mathcal{F}_m$, then m is known as *Bayesian* mass function.
- if there is only one element in $\mathcal{F}$ ($|\mathcal{F}| = 1$), then m is called a *categorical* mass function. This case is equivalent to a traditionally logical set.
- if focal elements in $\mathcal{F}_m$ form a chain in $\left(2^{\Omega}, \subseteq\right)$, then m is known as *consonant* mass function. This case is equivalent to a possibility distribution.

2.1.2 Plausibility, belief, and contour functions

Definition. Given a normalized mass function m on Ω, and $S \subseteq \Omega$, the **belief** and **plausibility** functions of S are defined respectively as following:

$$Bel(S) \stackrel{\text{def}}{=\!=} \sum_{S' \subseteq S} m(S'), \tag{2.3}$$

$$Pl(S) \stackrel{\text{def}}{=\!=} \sum_{S' \cap S \neq \emptyset} m(S') = 1 - Bel(\overline{S}). \tag{2.4}$$

Interpretation:

- $Bel(S)$ is a measure of the **total belief support** for S.
- $Pl(S)$ is the measure of the **lack of support** in $\overline{S}$ (or **consistency** with S).
- We always have: $Bel(S) \leq Pl(S)$.

The mass function, belief function, and plausibility function are mathematically equivalent. This means that if one of the three functions is determined, then the other two functions can be identified [14]:

$$m\big(S\big) = \sum_{S' \subseteq S} \big(-1\big)^{\big|S \setminus S'\big|} Bel\big(S'\big). \tag{2.5}$$

The **contour** function is the mapping $pl : \Omega \mapsto [0, 1]$ defined by

$$pl(\omega_i) = Pl\big(\{\omega_i\}\big), \forall \omega_i \in \Omega. \tag{2.6}$$

The basic probability assignment can be viewed as determining a set of the probability distribution m on the power set 2^{Ω} satisfying:

$$Bel(S) \leq Pl(S), \forall S \subseteq \Omega. \tag{2.7}$$

Thanks to the inequalities in (2.7), Pl and Bel are interpreted as the *upper* and the *lower* probability of S, respectively. These probabilities quantitatively reflect the incompleteness of the available evidence. If m is a Bayesian probability, then the inequalities in (2.7) degenerate to the equality ones.

2.1.3 Combination rule in evidence theory

Let m_1 and m_2 be two mass functions over the **same** frame of discernment Ω. Each m_i is derived from **one piece** of evidence. These two mass functions could be combined via the so-called Dempster's rule defined as following:

$$\big(m_1 \oplus m_2\big)\big(S\big) = \begin{cases} 0 & \text{if } S = \emptyset \\ \frac{1}{1-\kappa} \sum_{S' \cap S'' \neq \emptyset} \big[m_1\big(S'\big) \times m_2\big(S''\big)\big] & \text{if } S \neq \emptyset \end{cases} \tag{2.8}$$

where κ is the **degree of conflict** between two sources identified by:

$$\kappa = \sum_{S' \cap S'' = \emptyset} m_1\big(S'\big) \times m_2\big(S''\big). \tag{2.9}$$

In general, this rule could be repeatedly applied as many times as possible when there are more than two pieces of evidence. This advantage is useful for making decision. It implicitly assumes that two evident sources agree with each other in supporting a **common** proposition.

2.1.4 Decision making

Assume that we have a mass function m storing all knowledge about a random variable X. Our objective is to choose one predicted value $\omega_i \in \Omega$ for the random variable X. The mass function m can be converted into the so-called **pignistic probability distribution** defined by $BetPm : \Omega \mapsto \big[0, 1\big]$ [39], and:

$$BetPm(\omega_i) = \sum_{A \subseteq \Omega,\ \omega_i \in A} \frac{m(A)}{|A|}. \tag{2.10}$$

Basing on (2.10), the element ω_i with the highest value is selected as the prediction for X, i.e.,

$$X = \omega^* = \underset{\omega_i \in \Omega}{\operatorname{argmax}}\, BetPm(\omega_i). \tag{2.11}$$

The rule (2.11) yields a unique conclusion because it selects a single element in Ω with the highest value.

2.1.5 Example of decision making in DST

Assume that there are three keywords in user vocabulary called **pet**, **food**, and **sport**, denoted as ω_1, ω_2, and ω_3 respectively. In this case, the frame of discernment $\Omega = \{\omega_1, \omega_2, \omega_3\}$. Our objective is to compute the weight of

each keyword. Additionally suppose that there are **five pieces of evidence**, each of which is **somehow transferred** into the so-called mass functions as given in table 2.1. In that table, the mass m_3 is known as absolute certainty, m_1 is completely ambiguous (i.e., total ignorance), and m_2 totally refutes all words in Ω (i.e., open-world). The mass functions m_4 and m_5 correspond to situations of imperfect knowledge (m_4 is Bayesian). In this case, combining m_4 and m_5, then computing pignistic probability yields $BetP(\omega_1) = 0.09/(1-0.21)$, $BetP(\omega_2) = 0.60/(1-0.21)$, $BetP(\omega_3) = 0.10/(1-0.21)$, and $\kappa = 0.21$[1]. The value $\kappa = 0.21$ is the conflict degree between two sources m_4 and m_5. Basing on these weights, we can extract top-n keywords to form the user preferences, where n could be equal to 1 (only select ω_2), 2 (ω_2 and ω_3), or 3 (ω_2, ω_3, ω_1).

Table 2.1: An example of combination and decision making in DST.

$\mathcal{F}$	$m_1(\mathcal{F})$	$m_2(\mathcal{F})$	$m_3(\mathcal{F})$	$m_4(\mathcal{F})$	$m_5(\mathcal{F})$
$\emptyset$	0	1	0	0	0
$\{\omega_1\}$	0	0	0	0.3	0
$\{\omega_2\}$	0	0	0	0.6	0
$\{\omega_3\}$	0	0	1	0.1	0
$\{\omega_1, \omega_2\}$	0	0	0	0	0
$\{\omega_1, \omega_3\}$	0	0	0	0	0
$\{\omega_2, \omega_3\}$	0	0	0	0	0.7
Ω	1	0	0	0	0.3

2.2 Machine learning techniques

2.2.1 Overview of k-means clustering

Informally, k-means clustering is an approach for partitioning n observations into k distinct, non-overlapping partitions. Each partition is called a cluster, and is represented by a cluster center (usually referred to as a centroid). A **cluster** is a collection or group of "similar" data points according to certain criteria. To perform k-means clustering, practitioners must first specify a value for k - the desired number of clusters and the similarity criteria between two data points (e.g., Manhattan distance and Euclidean distance). Then the k-means algorithm will repeatedly assign each data point to the closest cluster. The algorithm is briefly described as below:

[1] In general, we have to fuse all masses into an overall one before decision making. Here we ignore m_1, m_2, m_3 because $m_1 \oplus m_i = m_i, \forall i$, $m_2 \oplus m_j = m_2, \forall j$, and $m_3 \oplus m_k = m_3, \forall k$

- Initial k cluster centers
- Repeat:
  - Assign each data point to the closest center
  - Recalculate centers as the mean of the points within a cluster
  - Stop if there is "no change"

Figure 2.1 and Figure 2.2 demonstrate the 3-means clustering on a set of 12 data points.

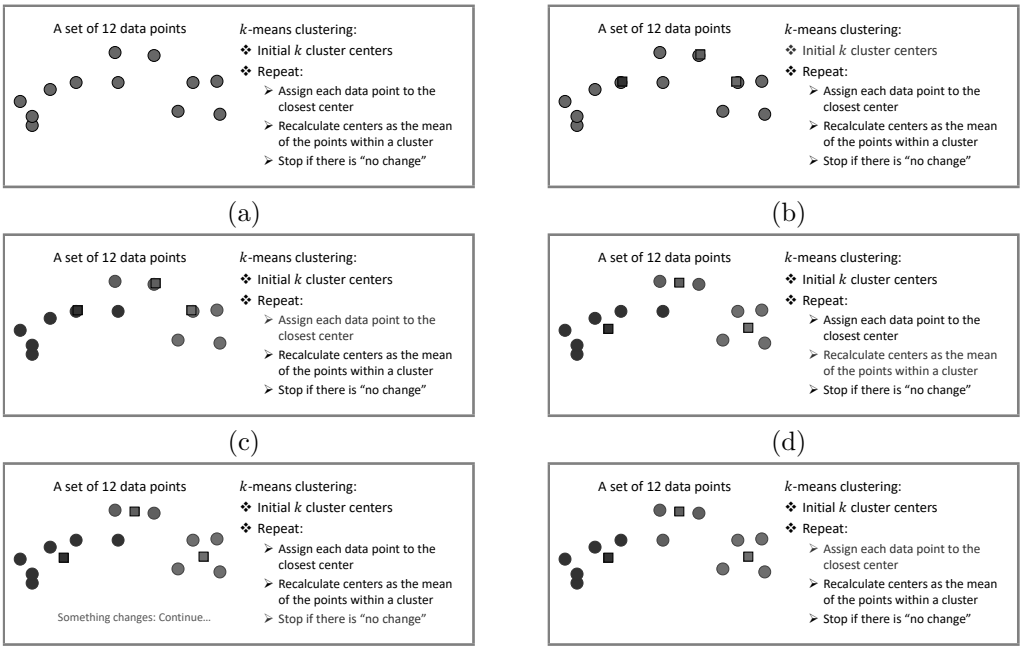


Figure 2.1: The demonstration of k-means clustering (1/2)

2.2.2 Deep generative neural networks

An Overview of Neural Networks. An artificial neural network (or a Neural Network for simplicity) can be defined as a computational model that "mimics" biological neural networks of the human brain in processing information. The elementary unit of the brain is a **biological neuron** as depicted in left part of Figure 2.3. One biological neuron consists of cell body, axon, dendrites, and synapses or "axon terminals". The human brain contains approximately 86 billion neurons, and they are connected by dendrites with

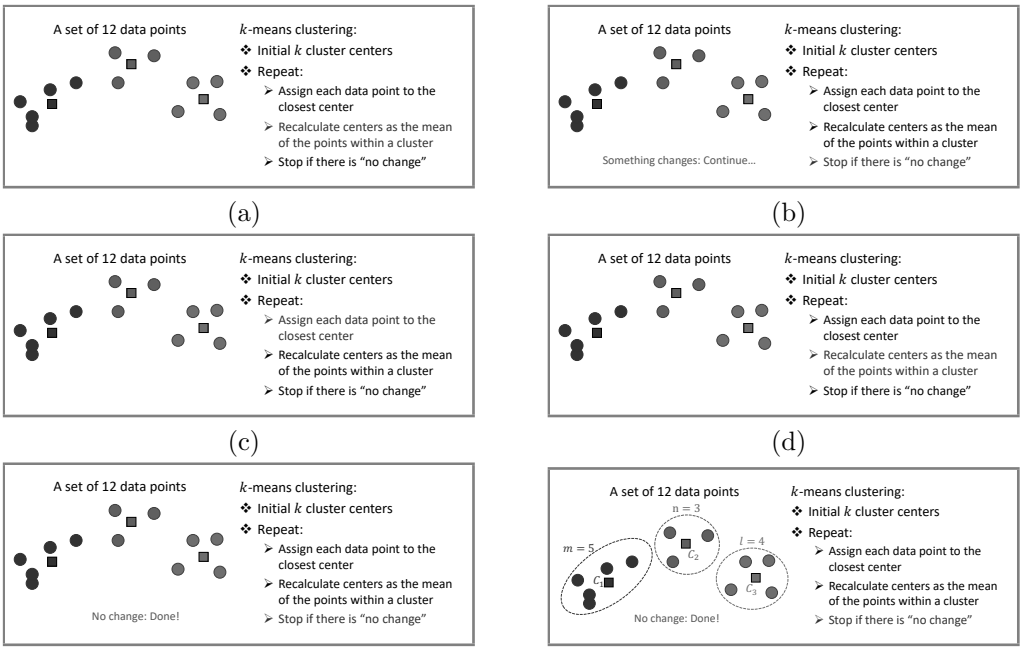
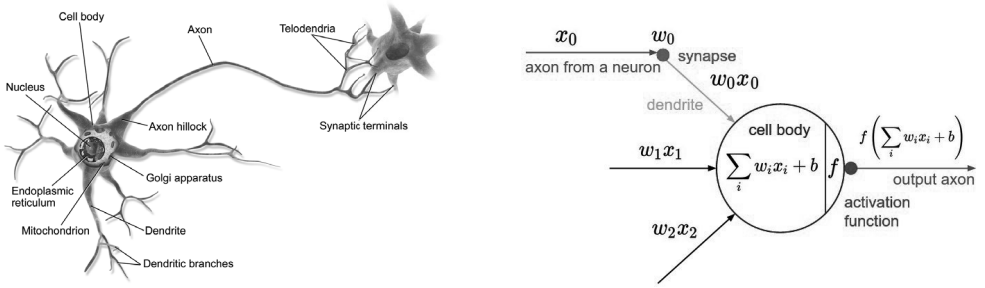


Figure 2.2: The demonstration of k-means clustering (2/2)



(a) The biological neuron, (source: wikipedia, keyword: Neuron)

(b) The simulated, computational neuron (source: Doctoral thesis of Andrej Karpathy).

Figure 2.3: The biological neuron and the computational neuron.

about $10^{14} - 10^{15}$ synapses [40]. The elementary computational unit of an artificial neural network is called a **node** corresponding to the cell body in a biological neuron. The right part of Figure 2.3 demonstrates a snapshot of one node. It receives input from some other nodes (or from an external source), carries out a computation process to produce an output. Each input has an associated weight (w), which is modeled as its relative significance level to other inputs. The node applies an activation function to the weighted

sum of its inputs to produce the output. This output then serves as input to other nodes through a **connection**. These connections forms a network architecture. The typical architecture of a network consists of input nodes (input layers), hidden nodes (hidden layers), output nodes (output layers), connections and weights, activation functions, and training rule (a mechanism defines the way the parameters of the neural network are updated during training process). A neural network that has more than one hidden layer is called a **deep neural network**. Figure 2.4 demonstrates a network architecture and its application on image classification. Researchers have proposed many kinds of efficient deep neural networks depending on a specific kind of input data, such as a Convolutional Neural Network (CNN) [41] to work with image data, or a Recurrent Neural Network [42] to deal with with textual data. In two recent decades, deep neural networks have brought remark-

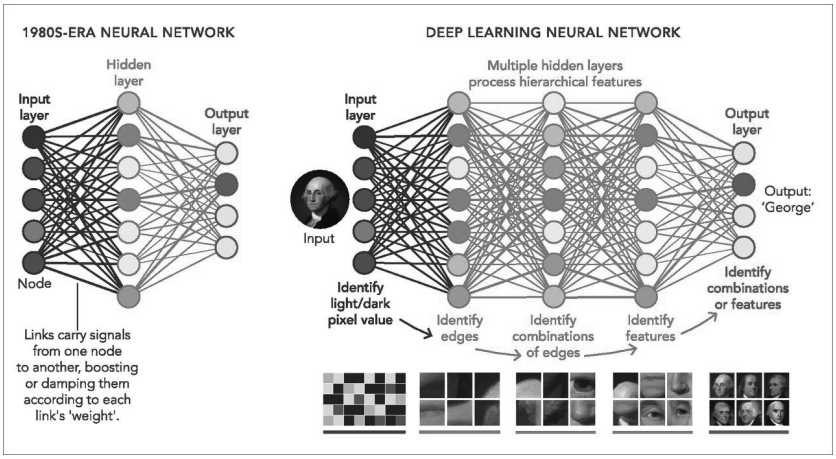


Figure 2.4: Neural network architectures [2].

able achievements in various machine learning problems, including supervised learning, semi-supervised learning, and unsupervised learning [7, 8, 9]. Practitioners have proposed many robust, data-driven models that can learn from input observations to make accurate predictions, find hidden patterns, or even create new instances that are very similar to the input data (e.g., generative models) [10, 11, 12]. Next paragraphs will briefly introduce two widely used generative models, the Variational Autoencoder and Generative Adversarial Network.

Variational Autoencoders. First introduced by Kingma and Welling in [11]; and Rezende et al. in [43], the VAE model and its variants ([44, 45, 46, 47, 48]) are the typical approaches for unsupervised learning of complicated

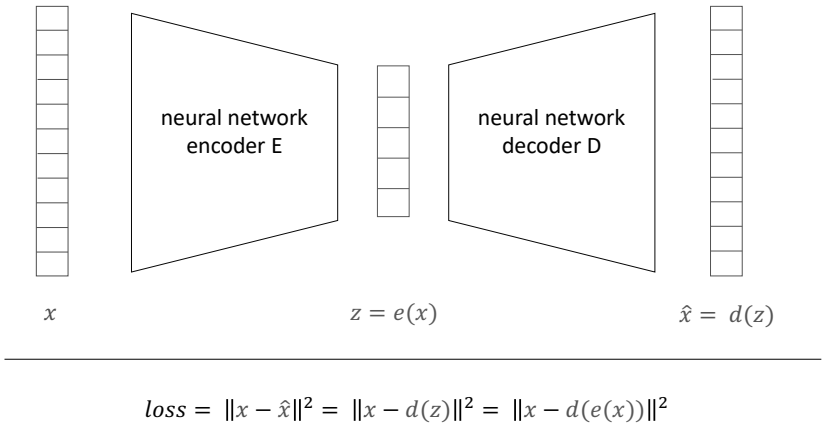


Figure 2.5: A typical architecture of an Autoencoder.

distributions. From the first launch, VAE has already demonstrated its efficient ability in generating many kinds of complicated data, including face images [11, 43, 49], handwritten digits [11, 50], CIFAR images [51], house numbers [12, 51], segmentation [52], physical models of scenes [49], and forecasting from static images [53].

Before describing Variational Autoencoders in details, it is necessary to define an **autoencoder**. An **AutoEncoder** (AE) is a special class of of neural networks, and is used for learning efficient data representations of unlabeled data in unsupervised learning. During the training process, AE ignores insignificant data ("noise") to focus on learning a representation of the data set. This representation is usually used for data **dimensionality reduction** in machine learning. Typically, an AE consists of two components, the encoder and the decoder. Each component corresponds to a neural network. The encoder compresses original data points from the initial space into the encoded space (also referred to as the latent space). The decoder reverses that process. These two components work in collaboration to learn a "good" encoding-decoding scheme through an iterative optimisation process. The objective function of AE is to minimze total loss during the encoding-decoding process over the entire data set. Technically, at each iteration a group of data instances is fed to the network (a encoder followed by a decoder), then the encoded-decoded instances (the output from the network) are compared with the original instances, and the errors are backpropagated throughout the network to update the parameters by gradient descent algorithm. Figure 2.5 depicts a typical architecture of an autoencoder. Informally, a VAE could be defined as an autoencoder in which the latent space in the encoder is regularised during the training process to be able to generate new instances. Additionally, the term "variational" arrives

from the close relation between two techniques in statistics, the regularisation and the variational inference.

The Objective Function of VAE. This section introduces some basic concepts that are necessary for mathematically deriving the optimization function of a VAE. The derivation primarily follows the work in [54].

Bayes' Theorem. This theorem provides a mathematical approach to update conditional probability of an event when new pieces of evidence are observed. The theorem is stated mathematically as below:

$$P(Y|X) = \frac{P(X|Y) \times P(Y)}{P(X)} \tag{2.12}$$

where X and Y are two events, $P(X) \neq 0$, and:

- $P(Y|X)$: the conditional probability that the event Y occurs, given that X is observed. $P(Y|X)$ is usually preferred to as the **posterior** probability of Y given X.
- $P(X|Y)$: the conditional probability that the event X occurs, given that Y is observed. $P(X|Y)$ is usually called the **likelihood** of Y given a fixed X because $P(X|Y) = L(Y|X)$.
- $P(X)$ and $P(Y)$ are the probabilities that the event X and the event Y are observed, respectively, without any given conditions. These two terms are preferred as the **prior** or **marginal probability**.
- X and Y are not identical events.

Bayes formula could be derived from the conditional probability as follow:

$$P(X|Y) = \frac{P(X \cdot Y)}{P(Y)}, \text{if } P(Y) \neq 0 \tag{2.13}$$

where $P(X \cdot Y)$ is the joint probability that both event X and Y are observed. Similarly,

$$P(Y|X) = \frac{P(X \cdot Y)}{P(X)}, \text{if } P(X) \neq 0. \tag{2.14}$$

Solving for $P(X \cdot Y)$ and substituting into the above equation for $P(Y|X)$ gives us the Bayes' formula:

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)} \tag{2.15}$$

Kullback-Leibler Divergence (KL divergence). This divergence is a measure to compare the density similarity between two distributions. It is the expectation of the difference of information content between the two distributions. Information content (or self-information, or Shannon information) is a quantity that reflects the level of "surprise" of a particular outcome from a random variable. Intuitively, an event with high probability contains lower information content. Given a random variable u, $p(u)$ and $q(u)$ are two distributions associated with u, then:

$$\text{The information content associated with } p(u)\text{: } IC_p(u) = -\log\left(p(u)\right) \tag{2.16}$$

$$\text{The information content associated with } q(u)\text{: } IC_q(u) = -\log\left(q(u)\right) \tag{2.17}$$

Therefore, the difference between information content associated with $p(u)$ and information content associated with $q(u)$ is,

$$\Delta IC = IC_p(u) - IC_q(u) = -\log p(u) + \log q(u) = \log\left(\frac{q(u)}{p(u)}\right). \tag{2.18}$$

The KL divergence is defined as the expectation of the above expression, i.e.,

$$\begin{aligned} KL\left(p(u)||q(u)\right) \overset{\text{def}}{=} \mathbb{E}_{\sim p}\left[\Delta IC\right] &= \int \left(\Delta IC\right) p(u) du \\ &= \int p(u) \log\left(\frac{p(u)}{q(u)}\right) du. \end{aligned} \tag{2.19}$$

Similarly,

$$\begin{aligned} KL\left(q(u)||p(u)\right) \overset{\text{def}}{=} \mathbb{E}_{\sim q}\left[\Delta IC\right] &= \int \left(\Delta IC\right) q(u) du \\ &= \int q(u) \log\left(\frac{q(u)}{p(u)}\right) du. \end{aligned} \tag{2.20}$$

Note that the KL divergence is not a metric because it does not satisfy the symmetric property, i.e,

$$KL(p(u)||q(u)) \neq KL(q(u)||p(u)). \tag{2.21}$$

This is because the expectation is computed with respect to $p(u)$ in $KL(p(u)||q(u))$ while the expectation is calculated with respect to $q(u)$ in $KL(q(u)||p(u))$. Additionally, the KL divergence always yields a non-negative value, i.e.,

$$KL\left(p(u)||q(u)\right) = \int p(u) \log\left(\frac{p(u)}{q(u)}\right) du \geq 0. \tag{2.22}$$

To prove this property, we utilize the inequality (2.23) in mathematics.

$$\log x < x - 1, \text{for } x > 0 \tag{2.23}$$

Therefore

$$\begin{aligned} -KL\left(p(u)||q(u)\right) &= \int p(u) \log\left(\frac{q(u)}{p(u)}\right) du \\ &\leq \int p(u)\left(\frac{q(u)}{p(u)} - 1\right) du \\ &= \int p(u)\frac{q(u)}{p(u)} - \int p(u)du \\ &= \int q(u)du - \int p(u)du \\ &= 1 - 1 = 0 \end{aligned}$$

which implies,

$$KL\left(p(u)||q(u)\right) \geq 0. \tag{2.24}$$

Optimization function of VAE. The objective of encoder in VAE is to approximately estimate the posterior distribution $q_\phi(z|u)$, where u is the observed data (or evidence), z is the latent variable, ϕ is a set of parameters representing the encoder network. Similarly, the decoder part in VAE approximately estimates the likelihood distribution $p_\theta(u|z)$, where θ is a set of parameters representing the decoder network. Output created by the encoder is a set of parameters forming the distribution in the latent space. This latent distribution is then sampled to serve as the input of the decoder. Figure 2.6 illustrates a typical scheme of a VAE in which the latent space are assumed to followed a bi-variate Gaussian distribution (i.e., the number of dimensions of z is 2). The KL divergence between the approximate distribution $q_\theta(z|u_i)$ and the real posterior distribution $p(z|u_i)$ is determined by equation (2.25):

$$KL\left(q_\phi(z|u_i)||p(z|u_i)\right) = -\int q_\phi(z|u_i) \log\left(\frac{p(z|u_i)}{q_\phi(z|u_i)}\right) dz \geq 0. \tag{2.25}$$

By applying the Bayes' rule to the equation (2.25), we get:

$$\begin{aligned} KL\left(q_\phi(z|u_i)||p(z|u_i)\right) &= -\int q_\phi(z|u_i) \log\left(\frac{p_\theta(u_i|z)p(z)}{q_\phi(z|u_i)p(u_i)}\right) dz \geq 0 \\ &= -\int q_\phi(z|u_i)\left[\log\left(\frac{p_\theta(u_i|z)p(z)}{q_\phi(z|u_i)}\right) - \log p(u_i)\right] dz \geq 0. \end{aligned} \tag{2.26}$$

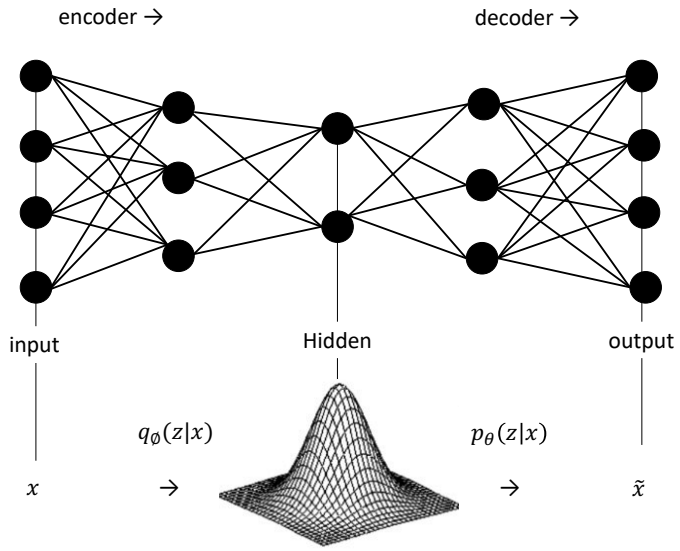


Figure 2.6: An typical scheme of an VAE with bivariate Gaussian distribution in the latent space.

Applying the distributive law into (2.26) yields,

$$-\int q_\phi(z|u_i)\log\left(\frac{p_\theta(u_i|z)p(z)}{q_\phi(z|u_i)}\right)dz+\int q_\phi(z|u_i)\log p(u_i)dz\geq 0. \tag{2.27}$$

Note that $\log(p(u_i))$ is just a constant, we can pull it out of the second integral, and $\int q_\phi(z|u_i)dz=1$ because $q_\phi(z|u_i)$ is a probability density function, the expression (2.27) becomes:

$$-\int q_\phi(z|u_i)\log\left(\frac{p_\theta(u_i|z)p(z)}{q_\phi(z|u_i)}\right)dz+\log p(u_i)\geq 0. \tag{2.28}$$

Equivalently,

$$\begin{aligned}\log p(u_i)&\geq\int q_\phi(z|u_i)\log\left(\frac{p_\theta(u_i|z)p(z)}{q_\phi(z|u_i)}\right)dz\\&=\int q_\phi(z|u_i)\Big[\log\left(p_\theta(u_i|z)\right)+\log p(z)-\log q_\phi(z|u_i)\Big]dz\\&=\int q_\phi(z|u_i)\log\left(\frac{p(z)}{q_\phi(z|u_i)}\right)dz+\int q_\phi(z|u_i)\log p_\theta(u_i|z)dz.\end{aligned} \tag{2.29}$$

Substituting the formula of KL divergence and Expectation into the right hand side of (2.29) yields,

$$\log p(u_i)\geq -KL\Big(q_\phi(z|u_i)\;||\;p(z)\Big)+\mathbb{E}_{\sim q_\phi(z|u_i)}\Big[\log p_\theta(u_i|z)\Big]. \tag{2.30}$$

The right hand side of (2.30) is called the Evidence Lower Bound (ELBO) (also preferred to as the variational lower bound). Because it is a lower bound of the log likelihood of the data ($\log p(u_i)$) that needs to be maximized. Therefore, maximizing the ELBO means looking for the largest value of the log likelihood of the evidence (our observed data). This is the key idea of **variational inference** because maximization of the log likelihood directly is usually an intractably computational task. The KL factor in the ELBO expression is called a regularizer because it imposes a constraint on the approximate posterior distribution $q_\phi(z|u)$ at the encoder. The expectation part in the ELBO is considered as the reconstruction term because it measures the likelihood of the reconstructed data produced by the decoder. Now, if we impose a Gaussian distribution onto the latent prior $p(z)$ and the approximate posterior $q_\phi(z|u_i)$, then a closed form for the loss function in VAE could be obtained.

Closed form of the objective function in VAE. Assume that we impose Gaussian distribution onto

$$p(z) \sim \frac{1}{\sqrt{2\pi\sigma_p^2}} e^{\left(-\frac{(z-\mu_p)^2}{2\sigma_p^2}\right)} \tag{2.31}$$

and

$$q_\phi(z|u_i) \sim \frac{1}{\sqrt{2\pi\sigma_q^2}} e^{\left(-\frac{(z-\mu_q)^2}{2\sigma_q^2}\right)}, \tag{2.32}$$

then the regularization term (KL) in the ELBO becomes:

$$\begin{aligned} &- KL\left(q_\phi(z|u_i) \;||\; p(z)\right) = \\ &\int \frac{1}{\sqrt{2\pi\sigma_q^2}} e^{\left(-\frac{(z-\mu_q)^2}{2\sigma_q^2}\right)} \times \log \left(\frac{\frac{1}{\sqrt{2\pi\sigma_p^2}} e^{\left(-\frac{(z-\mu_p)^2}{2\sigma_p^2}\right)}}{\frac{1}{\sqrt{2\pi\sigma_q^2}} e^{\left(-\frac{(z-\mu_q)^2}{2\sigma_q^2}\right)}} \right) dz. \end{aligned} \tag{2.33}$$

Applying the logarithm laws and simplifying the terms in the right hand side of (2.33), then expressing the result in term of Expectation yields,

$$\begin{aligned}
-KL\left(q_\phi(z|u_i) \mid\mid p(z)\right) &= \mathbb{E}_q\left[\log\left(\frac{\sigma_q}{\sigma_p}\right) - \frac{(z-\mu_p)^2}{2\sigma_p^2} + \frac{(z-\mu_q)^2}{2\sigma_q^2}\right] \\
&= \log\left(\frac{\sigma_q}{\sigma_p}\right) + \mathbb{E}_q\left[-\frac{(z-\mu_p)^2}{2\sigma_p^2} + \frac{(z-\mu_q)^2}{2\sigma_q^2}\right] \\
&= \log\left(\frac{\sigma_q}{\sigma_p}\right) - \frac{1}{2\sigma_p^2}E_q\left[(z-\mu_p)^2\right] + \frac{1}{2\sigma_q^2}\mathbb{E}_q\left[(z-\mu_q)^2\right].
\end{aligned} \tag{2.34}$$

Because the variance σ^2 is defined as the expectation of the squared distance from the mean, i.e.,

$$\sigma_q^2 = \mathbb{E}_q\left[(z-\mu_q)^2\right]. \tag{2.35}$$

Substituting (2.35) into (2.34) yields,

$$\begin{aligned}
-KL\left(q_\phi(z|u_i) \mid\mid p(z)\right) &= \log\left(\frac{\sigma_q}{\sigma_p}\right) - \frac{1}{2\sigma_p^2}\mathbb{E}_q\left[(z-\mu_p)^2\right] + \frac{\sigma_q^2}{2\sigma_q^2} \\
&= \log\left(\frac{\sigma_q}{\sigma_p}\right) - \frac{1}{2\sigma_p^2}\mathbb{E}_q\left[(z-\mu_p)^2\right] + \frac{1}{2} \\
&= \log\left(\frac{\sigma_q}{\sigma_p}\right) - \frac{1}{2\sigma_p^2}\mathbb{E}_q\left[(z-\mu_q+\mu_q-\mu_p)^2\right] + \frac{1}{2}
\end{aligned} \tag{2.36}$$

$$\begin{aligned}
&= \log\left(\frac{\sigma_q}{\sigma_p}\right) - \frac{1}{2\sigma_p^2}\mathbb{E}_q\left[(z-\mu_q)^2 + 2(z-\mu_q)(\mu_q-\mu_p) + (\mu_q-\mu_p)^2\right] + \frac{1}{2} \\
&= \log\left(\frac{\sigma_q}{\sigma_p}\right) - \\
&\frac{1}{2\sigma_p^2}\left\{\mathbb{E}_q\left[(z-\mu_q)^2\right] + 2\mathbb{E}_q\left[(z-\mu_q)(\mu_q-\mu_p)\right] + \mathbb{E}_q\left[(\mu_q-\mu_p)^2\right]\right\} + \frac{1}{2} \\
&= \log\left(\frac{\sigma_q}{\sigma_p}\right) - \frac{1}{2\sigma_p^2}\left\{\sigma_q^2 + 2\times 0\times(\mu_q-\mu_p) + (\mu_q-\mu_p)^2\right\} + \frac{1}{2} \\
&= \log\left(\frac{\sigma_q}{\sigma_p}\right) - \frac{\sigma_q^2 + (\mu_q-\mu_p)^2}{2\sigma_p^2} + \frac{1}{2}.
\end{aligned} \tag{2.37}$$

Therefore,

$$-KL\left(q_\phi(z|u_i) \;||\; p(z)\right) = \log\left(\frac{\sigma_q}{\sigma_p}\right) - \frac{\sigma_q^2 + (\mu_q - \mu_p)^2}{2\sigma_p^2} + \frac{1}{2}. \tag{2.38}$$

In the case of standard normal Gaussian distribution (i.e., $\mu_p = 0, \sigma_p = 1$), we get,

$$\begin{aligned} -KL\left(q_\phi(z|u_i) \;||\; p(z)\right) &= \log(\sigma_q) - \frac{\sigma_q^2 - \mu_q^2}{2} + \frac{1}{2} \\ &= \frac{1}{2}\log(\sigma_q^2) - \frac{\sigma_q^2 - \mu_q^2}{2} + \frac{1}{2} \\ &= \frac{1}{2}\left[1 + \log(\sigma_q^2) - \sigma_q^2 - \mu_q^2\right]. \end{aligned} \tag{2.39}$$

Recall that the ELBO in (2.30) is

$$\log p(u_i) \geq -KL\Big(q_\phi(z|u_i) \;||\; p(z)\Big) + \mathbb{E}_{\sim q_\phi(z|u_i)}\Big[\log p_\theta(u_i|z)\Big]. \tag{2.40}$$

With a specific data point u_i and a single random draw from the hidden space z, the right hand side of (2.40) reaches the maximum value at

$$\frac{1}{2}\left[1 + \log(\sigma_k^2) - \sigma_k^2 - \mu_k^2\right] + \mathbb{E}_{\sim q_\phi(z|u_i)}\Big[\log p_\theta(u_i|z)\Big] \tag{2.41}$$

where σ_k^2 and μ_k are parameters of the approximate distribution $q_\phi(z|x_j)$, and k is an index corresponding to the k^{th} component of the latent space vector $\vec{z}$. Therefore, the loss for a batch of data points is determined by,

$$-\mathcal{L}_{vae}(\phi, \theta) = \sum_{j=1}^{J} \frac{1}{2}\left[1 + \log\sigma_i^2 - \sigma_i^2 - \mu_i^2\right] + \frac{1}{M}\sum_{l} \log p\left(\vec{u}|\vec{z}\right) \tag{2.42}$$

where J is the dimensional size of the Gaussian latent $\vec{z} \sim \mathcal{N}(\vec{\sigma}, \vec{\mu})$, $q_\theta\left(\vec{z}|\vec{u}\right)$ is the approximate posterior, M is the number of samples randomly drawn regarding to the trick of reparametrization, and (ϕ, θ) are the parameters of the network to be optimized. The expression in (2.42) finishes deriving the loss function of VAE. Figure 2.7 summarizes the whole process of training a VAE using a batch of data points.

Generative Adversarial Networks. Generative Adversarial Network was introduced in 2014 by Ian J. Goodfellow et al. in the work entitled Generative Adversarial Nets [10]. A Generative Adversarial Network is an instance

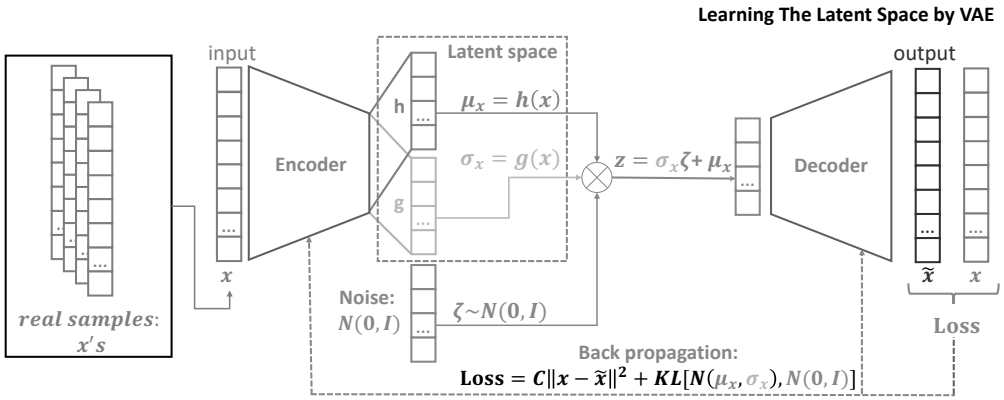


Figure 2.7: The training process in Variational Autoencoders.

of generative models. It consists of two neural networks, the generator and the discriminator. These two agents contest with each other in a zero-sum game. The gain of this agent is the loss of another agent and vise versa. Given a set of input data following a hidden distribution, the objective of the generator is to generate "fake" instances to lie the discriminator regarding to the distribution of given data. In contrast, the objective of the discriminator is to detect "fake" instances made by the generator. In term of loss value, the generator is trained to increase the error rate of the discriminator. The objective function of GAN is given by (2.43). Its derivation is provided concretely in the work [10].

$$\min_G \max_D \mathcal{L}_{gan}(D, G) = \mathbb{E}_{\vec{x} \sim p_{data}(\vec{x})} [\log(D(\vec{x}))] + \mathbb{E}_{\vec{z} \sim p(\vec{z})} [\log(1 - D(G(\vec{z})))] \tag{2.43}$$

where G is a differentiable function defined by a multilayer perceptron with parameters θ_g (θ_g is a set of weights that need to be trained in the perceptron G), known as the generator, and D is a second multilayer perceptron $D(\vec{x}; \theta_d)$ that outputs a single scalar (0 or 1) (θ_d is a set of weights that need to be trained in the perceptron D), known as the discriminator. $D(\vec{x})$ represents the probability that $\vec{x}$ came from the distribution learned from the observed data rather than "true" distribution p_G. The process of training a GAN model is carried out as follows:

- Noise from a random distribution is drawn, then fed into the Generator G to create the fake data x to produces the fake pair $(x, y = 0)$.
- The fake pair $(x, y = 0)$ and the real pair $(x, y = 1)$ are fed into the Discriminator D alternatively.

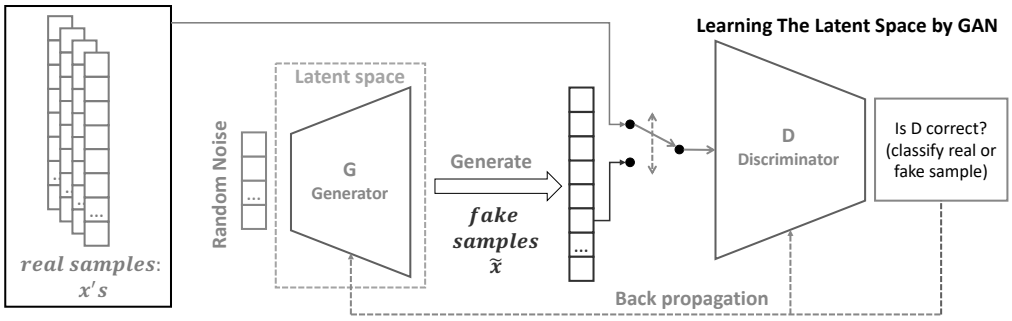


Figure 2.8: The training process in Generative Adversarial Networks.

- The discriminator D is a binary classifier network . It computes the loss for both fake x and real x, then combine these two losses as the final loss at D.
- Because G and D have different objective functions. Therefore the generator G also computes the loss from its noise as G loss.
- The two losses at G and D are backpropagated respective to their networks to update the parameters within its own networks.
- An optimization algorithm is utilized (e.g., Grad Descent, ADAM, or RMS prop) for training, and this process is iterated for a number of epochs or as long as the two networks are "good" enough in their learning task.

Although the Discriminator's loss is better than the Generator's loss in general. A good hint to stop training is that the losses of these two components are quite equivalent. This means that the Discriminator is extremely confused when deciding a "fake" or "real" on samples generated by the Generator. Figure 2.8 illustrates the training process of GAN model. GAN has been widely applied by research communities in many studies such as image generation [55, 56, 57, 58], anomaly detection for medical images [59], image inpainting or editing [60, 61, 62, 63], 3d object generation [64], anime characters or Emojis creation [65, 66], text to images [67, 68, 69, 70], and music generation [71].

2.3 Literature review

The research topic that is closely related to this research is **user profiling**, also referred to as expert profiling [72]. The approach for this topic

has increasingly attracted much attention after Craswell and his research team suggested the task of expert searching at Text REtrieval Conference Enterprise Track in 2005 [73]. The objective of this task is to extract an appropriate list of experts for a given queried topic. This is a sub-task of a more general problem called enterprise track. This enterprise track includes two tasks, the email search task and the expert search task. In that work, the authors aim at building an appropriate representation for the enterprise profile from a collection of heterogeneous documents. These collections could be obtained from a corporate intranet, email archives, and document repositories. Later, in [3] the authors introduced an approach for modeling expert profiles as a numerical vector. Each component of the modeled vector is expressed by a score value that reflects a specific skill of an expert. In addition, Balog et al. proposed a generative language modeling algorithm for the same problem [74]. Although these tasks are expected to work for the web track, the experiment is conducted on an internal enterprise data to mine the hidden relationships between entities inside the same organization. Recently, the increase in the amount of texts shared by users on social networking platforms requires more advanced methods which are able to cope with this kind of data. In [75], Estival et al. proposed a way of identifying author profiles by using texts extracted from English emails. However, this approach is challenging in practice because of user privacy issues. The problem is more challenging when input data are *short texts*, (e.g., a tweet on Twitter). There are taking short texts into account such as applying a neural network to classify gender based on manually labeled tweets [76], or seeking the set of most effective features for author identification problems by using messages extracted from Twitter [77]. Some other works used short texts to identify user occupation, infer basic demographic information [78, 79], detect the geographical location [80, 81, 82], learn user concern in politics and their intentions on election [83, 84], or infer the semantics of user profiles that are mainly used for improving the solution of the sentiment classification problem [85]. Most recently, the work in [86] uses tweets to infer the static profile of users on Twitter. Besides, in [4] the authors considered the problem of dynamic user profile for streams of short texts. In that work, they applied topic modeling for inferring the important terms in user vocabulary to form the final profile. However, this approach faces difficulties in capturing *new* topics which first appear anywhere in the timeline of user corpus. A summary on this research direction is that although these previous studies were designed for short texts, they considered different problem types, they did not consider the user preference problem, or did not consider the user preference problem under a dynamic context. If there exists some, then the proposed models are unable to efficiently generate fresh topics/tokens that have never appeared

in user vocabulary.

Another research approach is to consider the dynamics of user preferences as a **topic model**. A topic model is commonly viewed as a statistical model for inferring the distribution of virtual "topics" that are hidden inside a set of documents (corpus). Although researchers have introduced different approaches for the problem, the Latent Dirichlet Allocation-based model (LDA) [87, 35] is still one of the state-of-the-art approaches. LDA is a generative statistical model for textual data. It is claimed as the generalization of Probabilistic latent semantic indexing model (PLSI) [88]. The LDA model degenerates to PLSI model when the Dirichlet distribution of the prior is uniform [89]. In the assumption of the LDA model, each document is considered as a mixture of topics. Each topic is constituted by a set words. The main target is to infer the distribution of topics in each document and the distribution of words in each topic for a given corpus. Two typical approaches are Variational Inference [87] and Gibb Sampling [36]. Conjugate distribution pairs is important for the inferring process in LDA model, especially the Dirichlet-Multinomial conjugacy. This pairs allow the integral in the denominator of Bayesian theorem to be integrated out. Therefore, the inference process computationally feasible [90]. Besides, many dynamic topic models have been proposed for tracking the changes of topics in stream of short texts such as analyzing the evolution of topics over time via Gaussian distribution [35], dynamically tracking of single topic [91], reasoning the changes of consumer behavior over time [92]. There is a close relationship between text-based user preferences and the topic model. The user preferences problem could be solved by topic model-based approach, e.g., applying the LDA with collapsed Gibb sampling [87, 36, 93, 94] as our proposed one provided in Section 3.7. This idea has been used on literature such as emotion classification using short texts [95, 6, 96]. LDA and PLSI models work efficiently on long documents. However, their performances on short text data are not as efficient as expected because of feature sparsity issues[97]. Additionally, the work in [97] also provides a good survey about the problem of user preferences using the topic model. Taking short texts into consideration, Liang et al. applied topic model for the user profile problem [4, 98]. In these studies, the authors also considered the problem under a dynamic context which means that the profile changes over time. In our work, we define the problem in a similar manner but with a dramatically different approach. Overall, we applied deep generative networks for the purpose of learning and generating. Additionally, we use Dempster-Shafer theory of evidence for reasoning user preferences. The proposed approach is helpful for decision making, especially in recommendation systems. This is because the framework is able to generate a very *fresh* topic that a user may concern in

future while researchers in previous approaches face difficulties.

Chapter 3

The Integrated Framework of Learning and Reasoning for Static Profiles

3.1 Introduction

In the framework, word embedding technique, evidence theory, and k-means clustering are combined for learning the user profile under a static context. This framework is specifically designed for working with short texts created by social networking users, and dealing with uncertainty existing in each of user documents. Overall, the proposed framework consists of three main phases: (1) concept learning; (2) user modeling; (3) user profile extraction. Figure 3.1 intuitively provides an overview of the proposed framework. The main task in each phase is summarized as following:

Phase 1 - Concept learning: Abstract concepts are learned and represented at multiple levels of abstraction from user texts. At this phase, user texts are converted into word vectors via word embedding. Then abstract concepts are learned via k-means clustering. The learned concepts are organized in a hierarchical structure. Each concept reflects an appropriate semantic degree of user text which is helpful for reasoning user profiles.

Phase 2 - User modeling: Each document in user corpus is considered as one piece of evidence carrying some partial information that contributing to the user profile. These evident pieces are mathematically transferred into the so-called mass function in evidence theory by maximum a posterior estimation. Then Dempster's rule is utilized

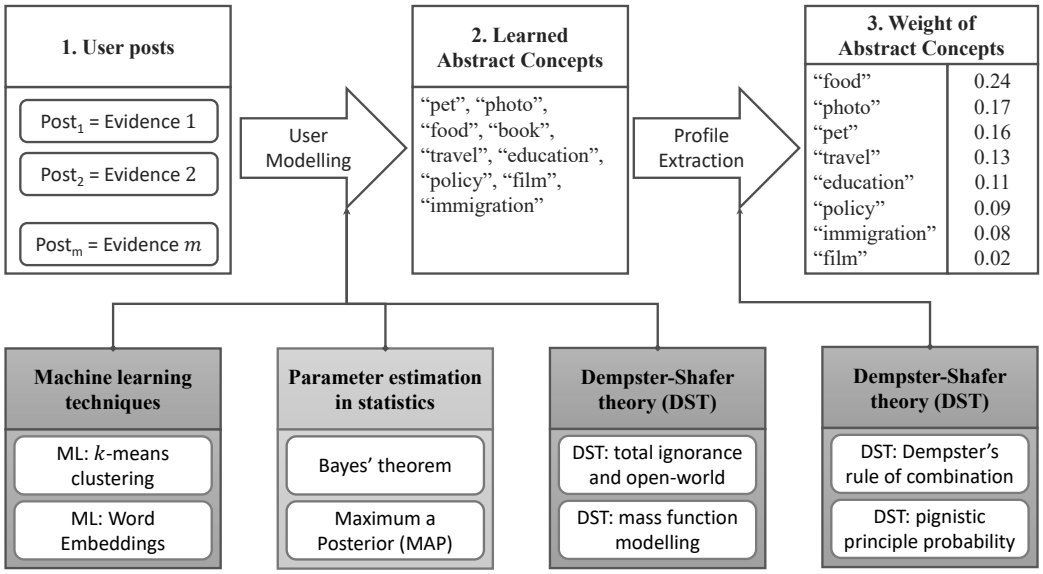


Figure 3.1: An intuitive viewpoint of the proposed framework for static profiles.

to combine all pieces of evidence into an overall mass function that represents for the entire corpus;

Phase 3 - Profile extraction: the so-called pignistic probability principle is used to compute the weigh of individual concepts in a set of abstract concepts learned from **Phase 1**. Then top-n concepts are are extracted to define the user profile.

Figure 3.2 provides a technical viewpoint of the proposed framework. Technical details will be described in the sections below.

3.2 Text collecting and preprocessing

One objective of the proposed framework is able to work with short texts made by users on social networks. Therefore, two data sets will be used are Twitter and Facebook. The raw data sets are processed through three main steps: tokenization, normalization, and noise elimination. Particularly, the following tasks will be carried out sequentially:

- White spaces, HTML tags, and special characters are removed.
- Stop words defined in the English word set are eliminated.

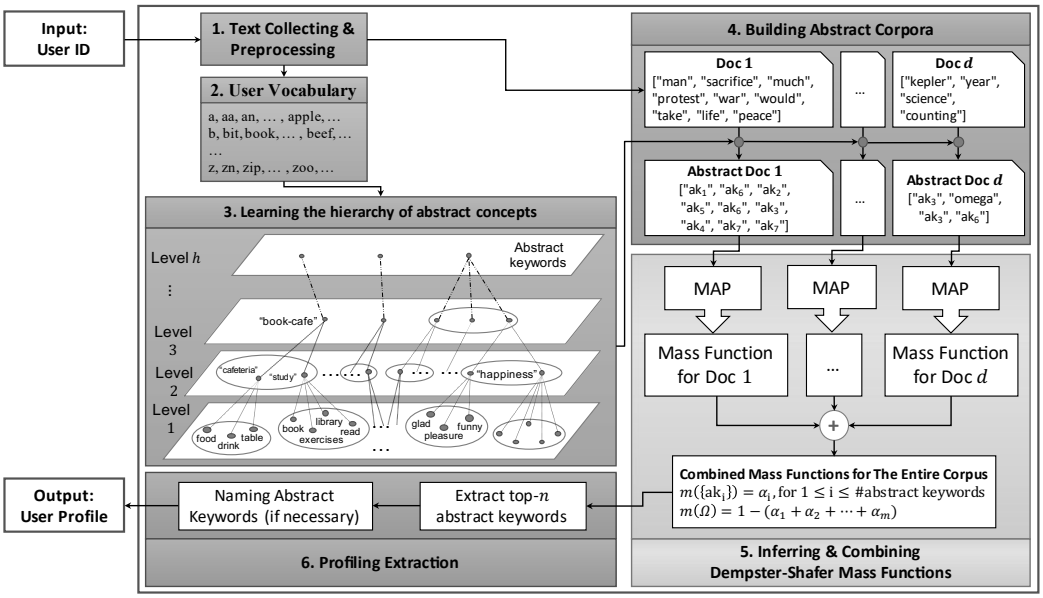


Figure 3.2: The proposed for the static profile using short texts.

- All remaining texts are converted into the corresponding lowercase.
- A stemming process is carried out to transfer a word into its original form. For example, the words "wolves" and "loving" are transferred into "wolf" and "love", respectively.

When this process is finished, two refined data sets are archived, the Facebook data set and the Twitter data set. These data sets are now ready to be used in the next step of the proposed framework for building the user vocabulary. Then this vocabulary is used for building the user model in the subsequent.

3.3 Learning the hierarchy of abstract concepts

This task is carried out by employing two machine learning techniques, the word embedding and the k-means clustering. Word embedding vectors store the semantics of words via the idea that "a word is characterized by the company it keeps"[1]. k-means clustering is used for learning and representing the hierarchy of abstract concepts. Each cluster is a representation of some

[1]This idea was first introduced by John Rupert Firth. Professor J. R. Firth is an English linguist and a leading scientist in British linguistics during the 1950s

hidden concepts shared by a group words in user vocabulary that are partially contribute to the user profile. Details of the two steps are described as follows.

Converting words into word vectors via a pretrained model In this phase, the pretrained word embedding technique, called Global model [99], is used to convert individual words into vectors. One reason of choosing GloVe model is that it was trained on Twitter data set. The training data set contains around two billion tweets with approximately 27 billion tokens. Weighted least squares regression was used as an objective function while training GloVe model. This function is defined as below:

$$\mathcal{L} = \sum_{j=1}^{|\mathcal{V}|} \sum_{k=1}^{|\mathcal{V}|} f\left(X_{jk}\right)\left(\theta_j^T e_k + c_j + c_k - \log X_{jk}\right)^2 \tag{3.1}$$

where $|\mathcal{V}|$ is the total number of tokens in the vocabulary, c_j, c_k are two bias terms, X_{jk} is the count number that the word j appears under the context of the word k, $f(X_{jk})$ is a weighting value, this value is equal to zero if $X_{jk} = 0$. There are many heuristics to select the value for f such that both frequent and rare co-occurrences are not over-weighted. After being trained, either e_k's, θ_j's, or the average value of these two vectors are possibly considered as the desired word vectors because these variables are exchangeable in the objective function (3.1).

Learning the hierarchy of abstract concepts by k-means clustering The objective of this task is to mine a set of abstract concepts that are hidden inside user plain texts. The learned concepts are represented into a hierarchical structure that is similar to the idea of ontology learning [100], [101], [102], [103]. This hierarchy explicitly provides the relationship between conceptual concepts. The lowest abstraction level in the hierarchy is used for representing plain texts. The words in a higher level are the abstract concepts. These abstract concepts are mined by k-means clustering algorithm. This hierarchical structure is visually depicted in Figure 3.3. Each abstract concept at a particular level generally captures some semantic features of the tokens at the lower level. For example, the words *glad*, *pleasure*, and *funny*, in a particular scenario, describe the similar user emotions. Therefore, these words could be represented by using one common abstract concept called "*happiness*" or "*good_feelings*", denoted as ak_i. This representation is useful for inferring user profiles because each abstract concept is assumed to store some pieces of conceptual information that partially contribute to the desired user profile. In practical application, this hierarchy is also necessary for

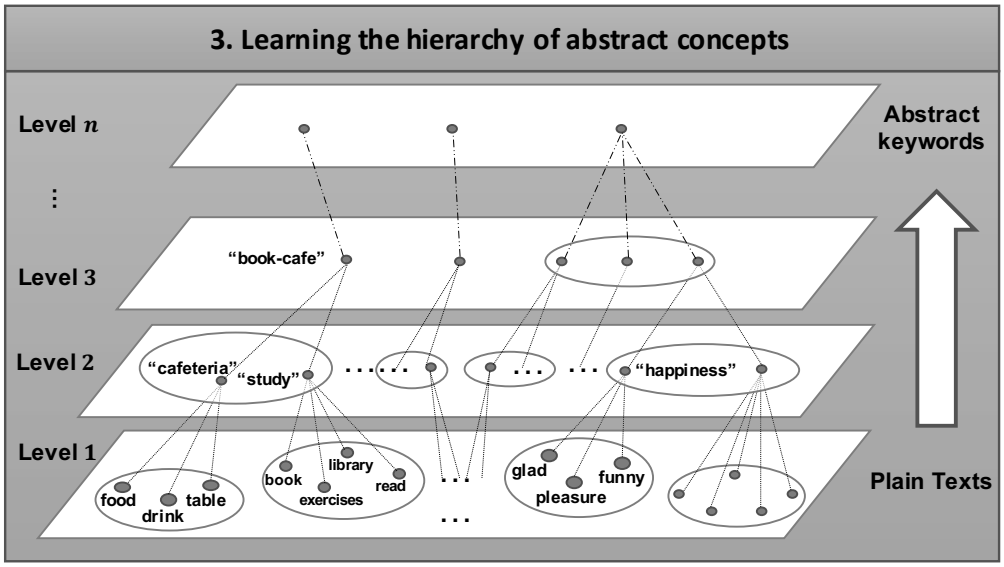


Figure 3.3: Learning the hierarchy of abstract concepts hide inside plain texts.

decision making. For instance, a web-based movie provider needs to estimate how much science fiction movies attract a specific user. This question can not be estimated directly via plain texts because there is no token in user observed keywords that exactly matches the top keywords in science fiction movies. Instead, the application can consider the similarity between these words in a more abstraction way. For example, the word "*virtual reality*" in some scenarios stores partial information about science fiction, although *fiction* and *reality* have opposite meaning in their plain forms. When this task is finished, a set of abstract concepts are obtained, called the abstract vocabulary. Each user has a corresponding conceptual hierarchy. Each abstract keyword in the hierarchy is represented by two terms, the abstract name and the corresponding word vector in $\mathbb{R}^n$ space ($n \in \mathbb{Z}^+$). With this representation, the framework could be able to utilize semantically statistical properties of words in user texts to infer the desired profile.

3.4 Building abstract corpora

For every document d in user corpus, the word w_i is replaced by its nearest abstract concepts basing on a similarity measure (e.g., cosine similarity or Euclidean distance). This process is provided in **Algorithm 1**. An extra keyword, called 'omega', is added into the set of abstract concepts. This

Algorithm 1: Building abstract corpus for a particular user

Input: text corpus $\mathcal{U}$, user dictionary, distant function d, threshold ϵ
Output: abstract corpus $\mathcal{U}'$ for the given user

```
1 for each document d ∈ U do
2     for each word w_i ∈ d do
3         get word vector of w_i
4         calculate all similarities sim(w_i, ak_j), ∀j
5         if all sim ≥ ε then
6             replaced w_i by 'omega' in d
7         else
8             replaced w_i by ak_j such that sim(w_i, ak_j) is largest
9 Return abstract corpus U'
```

addition is carried out at **step 5** of Algorithm 1. The word w_i is replaced by 'omega' if all similarities between w_i and abstract concepts ak_j ($\forall j$) exceed a predefined threshold ϵ. If this condition is satisfied, then it means that the framework is highly uncertain on which abstract concept the word w_i should be assigned. Therefore, it is reasonable to assigned w_i to all abstract concepts in the vocabulary set. This situation is interpreted as the *total ignorance* in evidence theory which is helpful for eliminating common words but not stop words in the abstract vocabulary (i.e., thing, object, people).

When this task is finished, an abstract corpus is obtained for each user. This corpus consists of abstract documents. Each document contains a list of "virtual" keywords, each of which stores some semantic concepts that are shared between concepts at a lower abstraction level according the hierarchy learned from the previous step. The next task is to quantitatively estimate the weight of individual concepts in the abstract vocabulary to infer the user profile.

3.5 Mass functions: derivation and combination

From now we use the term *documents (or corpus)* with means that *abstract documents (or abstract corpus)* for brevity. The user profiling problem now could be reformulated as below.

Reformulated Problem. Given a list of documents, each of which contained a list of abstract keywords, and $\mathcal{V} = \{ak_1, ak_2, ..., ak_k\} = \Omega$ is the set of all abstract keywords for a given user. The problem is to extract top-n keywords in $\mathcal{V}$ such that their weights are highest to form the user profile.

This reformulated problem could be broken into into sub-tasks:

> **Task a:** How to estimate the quantitative contribution of each document into the concepts in user vocabulary; and
>
> **Task b:** How to combine all quantities in the previous task into an overall one to assist the process of profile extraction.

In order to solve **Task a**, each document is considered as one piece of evidence carrying some partial information that contributes to the desired profile. Then each evident piece is mathematically transferred into the so-called mass function in Dempster-Shafer theory by maximum a posterior estimation. Each mass function is a quantitative measure of the corresponding document in user corpus. The solution for **Task b** could be found by utilizing Dempster's rule to combine all masses into an overall mass which is a representation for the entire corpus. This overall mass function provides a fundamental for the process of profile extraction in the final phase. Technical details on how to solve these two tasks are described as below.

Deriving Mass Functions.

- Let $\mathcal{V} = \{ak_1, ak_2, ..., ak_V, ak_{V+1} = \text{'omega'}\}$ be the set of all abstract keywords extracted at a specific level in the hierarchical structure.
- Consider a given set $\mathcal{W} = \{w_1,\ w_2, ..., w_N\}$ of N independent, identically distributed (i.i.d.) draws from a multinomial distribution on $\mathcal{V}$.
- In this case, $\mathcal{W}$ is considered as the document d created by a social networking user u_i, (e.g., a tweet on Twitter or a status on Facebook).

The likelihood of these drawings in the document d is computed by (3.2).

$$L(\vec{p}|\vec{w}) = p(\mathcal{W}|\vec{p}) = \prod_{i=1}^{N}\prod_{t=1}^{V+1} p_t^{[w_i=ak_t]} = \prod_{t=1}^{V+1} p_t^{n_t} \tag{3.2}$$

$$\sum_{t=1}^{V+1} n_t = N \text{ and } \sum_{t=1}^{V+1} p_t = 1, \tag{3.3}$$

where:

- ✓ n_t is the number of times abstract keyword ak_t was observed as a word in the document d (i.e., $\mathcal{W}$)
- ✓ Abstract keywords in $\mathcal{V}$ are assumed to follow a multinomial distribution, denoted as $Mult(ak_t \in \mathcal{V}|\vec{p})$
- ✓ Each component p_t in $\vec{p}$ is the probability that an abstract keyword ak_t is observed as a word w_i in a given document.

Applying Bayes' theorem to infer the posterior distribution yields,

$$p\big(\vec{p}|\mathcal{W},\vec{\alpha}\big) = \frac{\prod_{n=1}^{N} p(w_n|\vec{p})p(\vec{p}|\vec{\alpha})}{\int_{\mathcal{P}} \prod_{n=1}^{N} p(w_n|\vec{p})p(\vec{p}|\vec{\alpha})\, d\vec{p}} \tag{3.4}$$

where

$$\vec{p} \sim Dir(\vec{p}|\vec{\alpha}) = \frac{\Gamma\Big(\sum_{t}^{V+1} \alpha_t\Big)}{\prod_{t=1}^{V+1} \Gamma(\alpha_t)} \prod_{t=1}^{V+1} p_t^{\alpha_i - 1} \tag{3.5}$$

and $\vec{\alpha}$ is a concentration parameter vector which each element α_i corresponds to p_i in $\vec{p}$.

Because the denominator of (3.4) is a normalization factor, **maximum a posterior** estimation the right hand side of this equation means solving the constraint optimization problem defined by (3.6)

$$\arg\max_{\vec{p}} \quad \prod_{t=1}^{V+1} p_t^{n_t} \times \frac{\Gamma\Big(\sum_{t=1}^{V+1} \alpha_t\Big)}{\prod_{t=1}^{V+1} \Gamma(\alpha_t)} \prod_{t=1}^{V+1} p_t^{\alpha_i - 1} \tag{3.6a}$$

$$= \arg\max_{\vec{p}} \quad \prod_{t=1}^{V+1} p_t^{n_t + \alpha_t - 1} \times \frac{\Gamma\Big(\sum_{t=1}^{V+1} \alpha_t\Big)}{\prod_{t=1}^{V+1} \Gamma(\alpha_t)} \tag{3.6b}$$

$$\text{subject to} \quad \sum_{t=1}^{V+1} p_t = 1. \tag{3.6c}$$

Apply Lagrange multiplier method for solving this constraint optimization problem, we obtain,

$$p_t = \frac{n_t + \alpha_t - 1}{\sum_{t'=1}^{V+1} (n_{t'} + \alpha_{t'} - 1)}, \forall t \in [1, V+1]. \tag{3.7}$$

In words, the posterior distribution of an abstract concept ak_t is proportional to the frequency of that concept in the document d. The hyperparameter $\vec{\alpha}$ is considered ad the pseudo-count vector. Each component α_i

of that vector corresponds to the abstract keyword ak_i in the abstract vocabulary $\mathcal{V}$. An appropriate choice of the vector $\vec{\alpha}$ may vary depending on applications. In this work, a uniformly smoothed vector is assigned to all abstract keywords (i.e., Laplacian smoothing).

Now, applying (3.7) to compute the mass function associated with each document d, we obtain,

$$m_d\big(\{ak_t\}\big) = \frac{(\#\text{times } ak_t \text{ appears in } d) + \alpha_t - 1}{(\#\text{words in } d) + \sum_{t=1}^{V+1} \alpha_t - (V+1)}, \tag{3.8}$$

$$m_d(\Omega) = \frac{(\#\text{times 'omega' appears in } d) + \alpha_{omega} - 1}{(\#\text{words in } d) + \sum_{t=1}^{V+1} \alpha_t - (V+1)} \tag{3.9}$$

where $\{ak_t\} \subseteq \Omega = \big\{ak_1, ak_2, ..., ak_V\big\}$, $\forall t \in \big[1, V\big]$.
We verify that the total sum of all masses is equal to 1 $\big(m(\Omega) + m(\{ak_i\}) = 1, \forall i \in \big[1, V\big]\big)$. In the next step, Dempster's rule is used to combined all mass functions into an overall mass which is the representation for the entire corpus.

Combining mass functions. Let $\mathcal{L} = \{m_1, m_2, ..., m_L\}$ be a set of mass functions over the same frame of discernment Ω, where Ω is a set of all abstract concepts mined by k-means clustering plus the additional concept total ignorance **'omega'** ($\Omega = \{ak_1, ak_2, ..., ak_V, ak_{V+1} = 'omega'\}$). Each m_i is derived from the corresponding document d_i in user corpus via (3.8), (3.9).

If there is only two pieces of evidence ($L = 2$), then the Dempster's rule could be used for combining these two masses as defined by (3.10), (3.11).

$$\big(m_1 \oplus m_2\big)\big(\{ak_t\}\big) = \frac{1}{1-\kappa} \sum_{\{ak_t\}=S\cap S'} m_1(S) \times m_2(S'),\ (1 \le t \le V+1), \tag{3.10}$$

$$m(\Omega) = \frac{m_1(\Omega) \times m_2(\Omega)}{1-\kappa} \tag{3.11}$$

where κ is the conflict degree between two evident pieces identified by (3.12):

$$\kappa = \sum_{S\cap S'=\emptyset} m_1(S) \times m_2(S'). \tag{3.12}$$

In general, if there are many pieces of evidence $L > 2$, then the general formula for combining all evident pieces into an overall mass are provided in (3.13) and (3.14). In the **Appendix** section, we derive these equations by

inductive proof. The combination mass function is the representation for the entire user corpus.

$$m^{(1,2,..,L)}(\{ak_t\}) = \frac{1}{\mathcal{K}}\left(\prod_{i=1}^{L} m_i(\{ak_t\}) + \mathcal{C}\right), (1 \leq t \leq V+1) \tag{3.13}$$

$$m^{(1,2,...,L)}(\Omega) = \frac{\sum_{i=1}^{L} m_i(\Omega)}{\mathcal{K}} \tag{3.14}$$

$$\mathcal{C} = \sum_{k=1}^{L-1} \sum_{\substack{S_j \subseteq \{1,...,L\} \\ |S_j|=k}} \prod_{u \in S_j} m_u\big(\{ak_t\}\big) \prod_{\substack{v=1 \\ v \notin S_j}}^{V+1} m_v(\Omega) \tag{3.15}$$

$$\mathcal{K} = \sum_{t=1}^{V+1} \left[\prod_{i=1}^{L} m_i(\{ak_t\}) + \mathcal{C} + \prod_{i=1}^{L} m_i(\Omega)\right] \tag{3.16}$$

where $m^{(1,2,...,L)}\big(\{ak_q\}\big)$ is a mass value assigned for a given subset $\{ak_q\} \subseteq \Omega$, $\forall q \in [1, V]$, and $\mathcal{K}$ is known as the normalization factor. This resulting mass function is the basis for extracting top-n concepts to form the user profile in the final phase of the proposed framework.

3.6 Profile extraction and abstract concept naming

In this phase, basing on the overall mass function in the previous step, the so-called pignistic probability principle defined in (2.10) is used for computing the weight of all singletons in Ω. Then, top-n concepts with highest probability are extracted to define the user profile via (2.11). **Algorithm 2** summarizes the entire process. **Algorithm 2** outputs the profile in two formats, the semantic/abstract profile and the actual profile. The semantic profile contains a list of abstract keywords (i.e., only word vectors of the abstract keywords). The actual profile is achieved by naming all abstract keywords to their nearest tokens in user vocabulary according to cosine similarity. Both formats are helpful depending on contexts. If an application concerns on the conceptual abstraction of user references, then abstract keywords are useful. If we care about what actual keywords that approximately best reflect user preferences, then the actual keyword profile is helpful. In practice, the semantic profile is quite more meaningful for decision making. For instance, a job offering application, or an recommendation agent wants to ***quantitatively*** estimate the concerning level of a given user on three topic entertainment, technology, and education. Assuming that the top-keywords

Algorithm 2: The integrated framework of learning and evident reasoning for static user profiles

Input: a positive integer k, text corpus, distance function d
Output: top-k abstract/actual keywords

Phase 1:
  Get word vector of words via pretrained GloVe model
  Learn the hierarchy of abstract concepts
Phase 2:
  Build abstract corpus: replace individual words by its nearest centroid
  Infer mass function for each document via (3.8), (3.9)
  Combine all mass functions via (3.13), (3.14),(3.15), (3.16)
Phase 3:
  Compute pignistic probability for all singletons via (2.10)
  Sort pignistic probabilities in descending order
  Pick up top-k abstract keywords with highest pignistic probability, called set S
  Name the abstract keywords if necessary, called S'
Return S/S' as user profile
End Algorithm.

on each topic are suggested by experts. Then, cosine similarity between the user semantic profile and the top-keywords in each topic could be computed to satisfy this requirement. In this case, the exact match becomes unfeasible and meaningless because of highly zero probabilities in the results.

3.7 Empirical Results

This section introduces the data sets, experiments, and evaluation metrics for comparing the efficiency between the proposed framework and baseline models.

3.7.1 Data sets

Two data sets are used in the experiments are Twitter[2] and Facebook[3]. The Facebook data set contains **1180** users after being processed. All posts in

[2]Available at https://bitbucket.org/sliang1/uct-dataset/get/UCT-Dataset.zip [98].
[3]This data set is collected by our research team under the policy of Facebook.

user timeline are crawled from the date of signing up the account to March 30, 2020. On average, each user uploaded approximately **620** posts, each of which contains about **36** words. Similarly, the Twitter data set contains **1020** users after being processed. All tweets in user timeline are collected from the date of opening account to May 31, 2015. In average, each user create around around **3219** tweets, each of which consists of approximately **12** tokens. The vocabulary size of both data sets is around **6068**.

The ground true keywords is achieved by considering the hashtags is user posts. Particularly, an automatic process is implemented as following:

1. extract the hashtags from user posts
2. sort these hashtags by their frequency in descending order
3. eliminate the symbol '#' in each hashtag
4. lowercase all capital letters
5. leave the resulting hashtags as ground truth keywords.

For instance, the crawled hashtag "#HumanRights" is transformed into "humanright" and this hashtag is then used for evaluation.

3.7.2 Baselines

Baseline models used in the experiments are described as below:

- GSDMM: this is a Dirichlet Multinomial Mixture model with a collapsed Gibbs Sampling technique for clustering short texts. The model is proposed by Yin *et al.* in 2014 [104].
- Rake: the Rapid Automatic Keyword Extraction model proposed by Rose et al. in 2010 for text description [105].
- TextRank: this is a graph-based model for summarizing text proposed in 2016 [106, 107].
- TFIDF: this is a traditionally statistical model that estimate how important a keyword is to a document in a corpus[4] [108].
- LDA: the Latent Dirichlet Allocation model uses a Gibb Samping technique to topic modelling. We based on this model to derive the algorithm for user profiling problem as shown in **Algorithm 4**.

[4]https://en.wikipedia.org/wiki/Tf–idf

- The proposed methods: Evidence theory-based user profiling. For each user, the proposed framework outputs two profiles, the semantic profile (dst_ab) and the actual profile (dst).

The **Algorithm** 3 depicts an approach for user profiling basing on TFIDF, RAKE, and TextRank models. Similarly, the **Algorithm** 4 provides the approach for LDA-based user profiling.

Algorithm 3: TFIDF/RAKE/TEXTRANK - based user profiling algorithm

Input: n - a positive integer (the profile length)
Input: Preprocessed text corpus
Output: top-n keywords representing the user profile

```
Merge all posts from user corpus into one document
Split the one document into n documents
Compute score for words in n documents
Let S = ∅
while |S| ≤ n do
    for each document d do
        extract the word w with highest score
        remove w from the d
        if w ∉ S then
            insert w into S
Return S
End Algorithm.
```

3.7.3 Experimental criteria and evaluation metrics

We conduct the experiments to evaluate the following criteria:

CR1. How is the overall performance of all models in capturing actual keywords?

CR2. How is the overall performance of all models in capturing the abstract concepts?

CR3. How does the size of word vectors affect the overall performance of all models?

CR4. What is the time complexity of all models when being applied for practical data sets?

Algorithm 4: LDA/GSDMM - based user profiling algorithm

Input: n - a positive integer (the profile length
Input: Preprocessed text corpus
Output: top-n keywords representing the user profile

```
Build LDA/GSDMM model with n topics
Let S = ∅
while |S| ≤ n do
    for each topic-term distribution do
        extract the term t with highest score
        remove t from the topic-term
        if t ∉ S then
            insert t into S
Return S
End Algorithm.
```

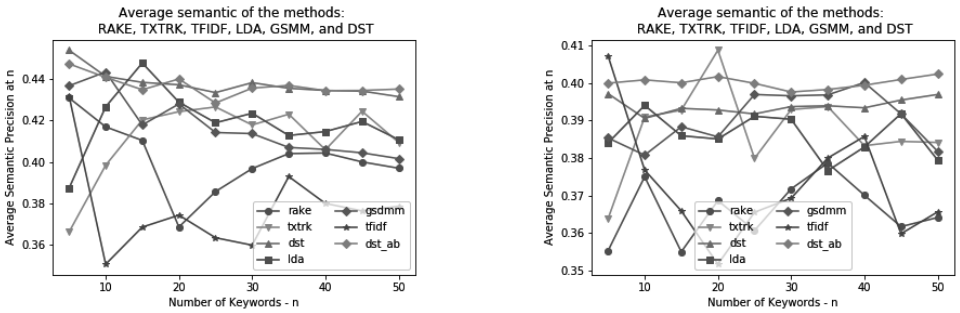
The **CR1** can be evaluated by using standard precision metric. The standard precision score is 1 if w_* and w_{gt} are identical, 0 otherwise, where w_* is a keyword in the extracted profile and w_{gt} is a ground truth keyword. The **CR2** can be evaluated via semantic precision (s-precision). The semantic precision is defined as the cosine similarity between the word vector $e(w_{gt})$ of the ground truth keyword w_{gt} and the word vector $e(w_*)$ of the retrieved keyword w_*. This score may provide some useful insights for decision making in many contexts such as recommendation systems. The **CR3** is evaluated by adjusting the size of word vectors to different values (e.g., 25, 50, 100, or 200) [99]. And the **CR4** can be evaluated by estimating executive time of all models when being applied on practical data sets.

3.7.4 Results

In all experiments, the value of hyper-parameters is set as follows: the abstraction level is 2, the distant function = *cosine similarity*. For each user, the profile length is varied from 5 to 50 with the a step size of 5. The preprocessed data sets are partitioned into train set and test set with ratio 7/3. The train set is used to tune for the best value of threshold ϵ and number of abstract keywords c ($\epsilon = 0.8$, and $c = 50$). The reported results are average values obtaining from 10 times of running the same experiments. Figure 3.4 to Figure 3.8 illustrate the average performance on various metrics of all models, including TextRank, RAKE, TFIDF, LDA, GSDMM, and the two

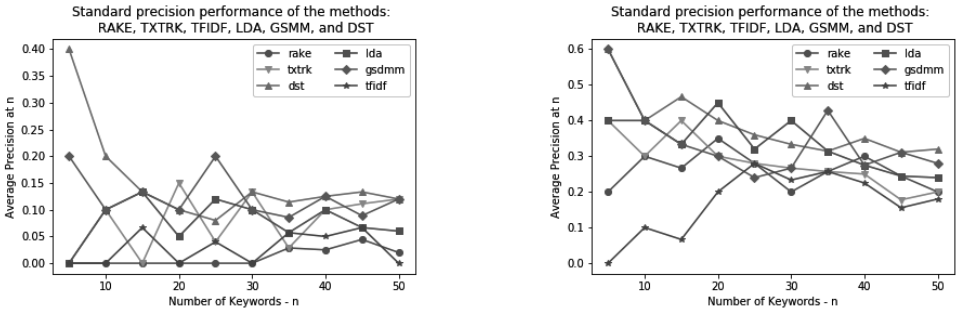
proposed models (*dst_ac* and *dst_ab*).

Standard Precision and Semantic Precision. **Figure 3.4** and **Figure 3.5** respectively provides the average performance of all models on Facebook and Twitter data sets. According to these figures, the proposed models (*dst* and *dst_ab*) outperform all baselines on both standard precision metric and semantic precision metric at many values of the profile length, especially at lengths 10, 15, 40, 45, and 50 on both data sets. These results confirm the efficiency of the proposed model, and validate the **CR1** and **CR2**. Additionally, the fluctuation of the proposed model (*dst_ab*) is more stable in comparison with the baselines on when the profile length is changed.



(a) Twitter data (b) Facebook data

Figure 3.4: The average performance on semantic precision of all models.



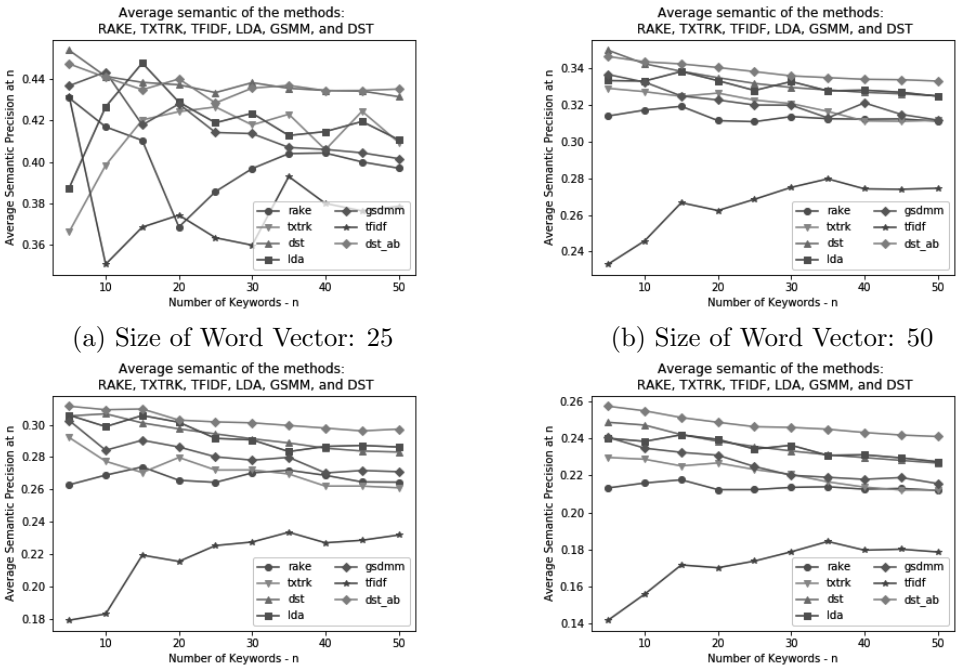
(a) Twitter data (b) Facebook data

Figure 3.5: The average performance on standard precision of all models.

The impact of word vectors' size. The size of word embedding vectors is adjusted to different values (25, 50, 100, and 200) for checking the impact of this adjustment on the performance of all models. The semantic precision is recorded for comparison. **Figure 3.6** and **Figure 3.7** illustrate the results on Twitter and Facebook data sets. Inspecting these figures reveals some observations as below:

- ✓ All models work most efficiently in capturing semantic concepts in user texts when the word vector takes size 25.
- ✓ The rank in performance of all models is nearly identical when the size of word vectors is changed. This rank could be classified into three categories in a descending order as follow: (dst_ab, dst) > (LDA, GSDMM, TextRank) > (RAKE, TFIDF).

This experiment again confirms the stability of the proposed model in the case of changing word vector size.



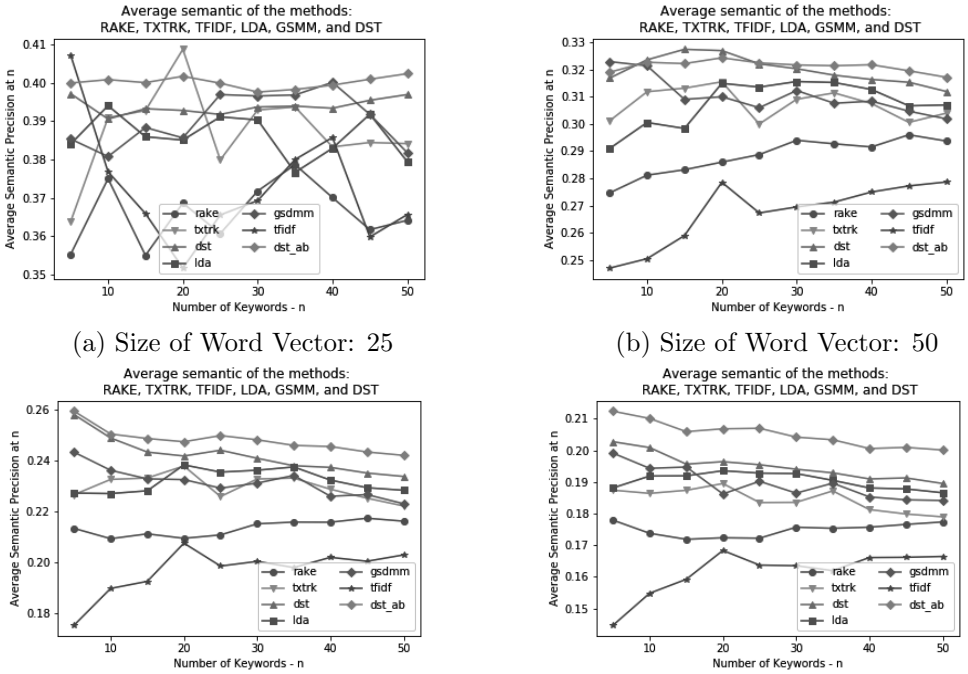
(a) Size of Word Vector: 25

(b) Size of Word Vector: 50

(c) Size of Word Vector: 100

(d) Size of Word Vector: 200

Figure 3.6: The performance on semantic precision of all models on Twitter data set when the word vector size is changed (200, 100, 50, and 25).



(a) Size of Word Vector: 25

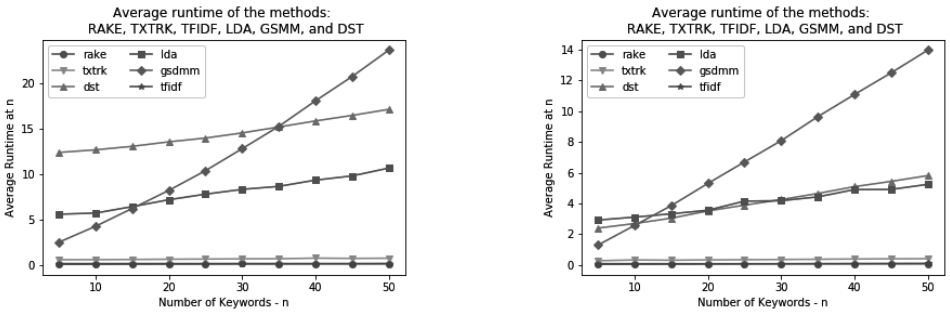
(b) Size of Word Vector: 50

(c) Size of Word Vector: 100

(d) Size of Word Vector: 200

Figure 3.7: The performance on semantic precision of all models on Facebook data set when the word vector size is changed (200, 100, 50, 25).

Time Complexity Comparison. This experiment will compare the executive time of all models in practical data sets. A Python built-in library was used for estimating the runtime of all models. The runtime is the difference between the start time when a built-in function is called and the end time when the called function is finished. In *dst* model, two types of executive time is recorded, the time required for user corpus preprocessing and the time required for profile extraction. The former type can be eliminated in comparison because it is executive only once per user. The later type is reported in the results. Figure 3.8 shows the average runtime of all models on both Facebook and Twitter data sets. As illustrated in that figure, dst model gives the second worst performance on runtime when being compared with baselines. This is one of the disadvantages of the proposed framework because evidence theory considers all subsets of a given set when modelling the user profile. In the worst case, the time complexity of the proposal is proportional to exponential functions ($O(2^n)$, where n is the number of element in the frame of discernment Ω).



(a) Twitter data (b) Facebook data

Figure 3.8: The runtime performance of all models.

3.7.5 Why does the proposed framework work quite well on short texts?

Baseline models, such as LDA or GSDMM, essentially base one the frequency of keywords in user vocabulary to compute the weight of individual term while inferring the user profile. For example, the posterior of topic assignment of each word z_i in collapsed Gibb Sampling-based LDA model ([87], [36]) is estimated by (3.17)

$$p\Big(z_i = k \big| \mathcal{Z}^{-i}, \mathcal{X}, \vec{\alpha}, \vec{\beta}\Big) \propto \frac{v_{k,w_{d,n}+\beta_{w_{d,n}}}}{\sum_i v_{k,i+\beta_i}} \times \frac{n_{d,k} + \alpha_k}{\sum_{i=1}^{K} n_{d,i} + \alpha_i} \tag{3.17}$$

where

- ✓ $v_{k,w_{d,n}}$ is the number of times the word type $w_{d,n}$ assigned to topic k,
- ✓ $n_{d,k}$ is the number of times the topic k appears in the document d,
- ✓ $\vec{\beta}$ is the Dirichlet parameter vector for word distribution within a topic,
- ✓ $\vec{\alpha}$ is the Dirichlet parameter vector for topic distribution within a document,

The first factor of (3.17) estimates to how much a topic β_i "likes" the word type $w_{d,n}$. The second factor estimates how much a document d "likes" the topic k. In words, these two factors indicate that the more frequent a keyword with assigned topic appears in user documents, the higher opportunity that keyword is chosen as a candidate for the user profile. This is reasonable and performs well for long texts. However, for short or medium size texts, this approach faces difficulties. This is because all terms in user short texts usually converge a uniformed distribution. This fact causes troubles for the

frequency-based models while sampling and choosing one specific term with highest frequency. In order to defeat this challenge, the proposed method replaces every individual keyword by a nearest abstract concept mined from user texts. Therefore, this replacement is helpful when deriving the so-call mass function in Dempster-Shafer theory by maximum a posterior estimation. **Example 2** below demonstrates how this process works on a simple user corpus.

Example 2. Assume that there is a simple corpus created by a Twitter user. This corpus contains four documents as provided in Table 3.1. The frequency of each keyword w_i in $\mathcal{V}$ is equal to 1 for the entire corpus. Therefore, the formula (3.17) faces difficulties in sampling and selecting the keywords with highest posterior probability. The proposed model, however, can overcome this challenge by considering the frequency of abstract concepts instead (**Table Table** 3.2). Particularly, the words *puppy, parrot, kitty, dog* are viewed as a abstract concept called "pet", denoted as 'ak_1'. The words *look, see, style* is replaced by a concept called *"picture"* or *"photo"*, denoted as 'ak_2. Similarly, other groups of similar plain texts are represented by a unique corresponding abstract concept (e.g., *"food"* is used as a representation of *hungry, eat, whistle*, denoted as 'ak_3'). Consequently, these abstract concepts, ak_1, ak_2, and ak_3 are more frequent in compared to other concepts in $\mathcal{V}$. Therefore, the weight associated with these concepts is high when applying maximum a posterior estimation for deriving the associate mass function for each document. Consequently, these abstract keywords are top candidates for profile extraction. Details of calculation process are demonstrated in **Table** 3.3 and **Table** 3.4.

Table 3.1: The demonstration of the proposed framework on a simple user corpus (1/4).

Doc	**Step 1. Original Texts**
1	Look at the style of my puppy when eating. How lovely he is!
2	Nothing better than starting a day with your parrot's whistle
3	See how this kitty is extremely clever in her own way
4	My dog is always hungry although eating a lot
	Step 2. Texts after being preprocessed
1	[look, style, puppy, eat, love]
2	[good, start, day, parrot, whistle]
3	[see, kitty, extreme, clever, way]
4	[dog, hungry, eat, lot]

Table 3.2: The demonstration of the proposed framework on a simple user corpus (2/4).

Step3. User Vocabulary	**Step 4. Building Abstract Corpora**
[dog, kitty, parrot, puppy,	[dog, kitty, parrot, puppy]: 'pet', denoted as 'ak_1'
look, see, style,	[look, see, style]: 'photo', denoted as 'ak_2'
eat, hungry, whistle,	[eat, hungry, whistle]: 'food', denoted as 'ak_3'
extreme, lot,	[extreme, lot]: 'degree', denoted as 'ak_4'
start, day,	[start, day]: some concept, denoted as 'ak_5'
way,	[way]: some concept, denoted as 'ak_6'
clever, good,	[clever, good]: some concept, denoted as 'ak_7'
love]	[love]: some concept, denoted as 'ak_8'

Table 3.3: The demonstration of the proposed framework on a simple user corpus (3/4).

Doc	**Step 5. Building Abstract Corpus**
1	['photo', 'photo', 'pet', 'food', 'ak_8']
2	['ak_7', 'ak_5', 'ak_5', 'pet', 'food']
3	['photo', 'pet', 'ak_4', 'ak_7', 'ak_6']
4	['pet', 'food', 'food', 'ak_4']
	6. Each document is considered as one piece of evidence contributing to the weight of abstract concepts
1	$m(ak_1) = 0.13$, $m(ak_2) = 0.16$, $m(ak_3) = 0.13$, $m(ak_4) = 0.09$, $m(ak_5) = 0.09$, $m(ak_6) = 0.09$, $m(ak_7) = 0.09$, $m(ak_8) = 0.13$, $m(\Omega) = 0.09$ (Ω is the set of all terms in user vocabulary)
2	$m(ak_1) = 0.13$, $m(ak_2) = 0.09$, $m(ak_3) = 0.13$, $m(ak_4) = 0.09$, $m(ak_5) = 0.16$, $m(ak_6) = 0.09$, $m(ak_7) = 0.13$, $m(ak_8) = 0.09$, $m(\Omega) = 0.09$
3	$m(ak_1) = 0.13$, $m(ak_2) = 0.13$, $m(ak_3) = 0.09$, $m(ak_4) = 0.13$, $m(ak_5) = 0.09$, $m(ak_6) = 0.13$, $m(ak_7) = 0.134$, $m(ak_8) = 0.09$, $m(\Omega) = 0.09$
4	$m(ak_1) = 0.129$, $m(ak_2) = 0.16$, $m(ak_3) = 0.097$, $m(ak_4) = 0.129$, $m(ak_5) = 0.097$, $m(ak_6) = 0.097$, $m(ak_7) = 0.097$, $m(ak_8) = 0.097$, $m(\Omega) = 0.097$

Table 3.4: The demonstration of the proposed framework on a simple user corpus (4/4).

Step 7. Fusing all pieces of evidence into an overall mass
$m(ak_1) = 0.17, m(ak_2) = 0.18, m(ak_3) = 0.12, m(ak_4) = 0.11,$ $m(ak_5) = 0.11, m(ak_6) = 0.10, m(ak_7) = 0.11, m(ak_8) = 0.10,$ $m(\Omega) = 0.005$
Step 8. Calculating the weight for individual abstract keywords and Extracting top-3 keywords to define the profile
$m(ak_1) = 0.17, m(ak_2) = 0.18, m(ak_3) = 0.12, m(ak_4) = 0.11,$ $m(ak_5) = 0.11, m(ak_6) = 0.10, m(ak_7) = 0.11, m(ak_8) = 0.10$
The extracted profile is: [**"pet", "photo", "food"**]

Chapter 4

Deep Generative Networks Paired with Evidential Reasoning for Dynamic Profiles

4.1 The proposed framework: an overview

The proposed framework combines advancements of deep neural networks and evidence theory for dynamically capturing user preferences. It consists of three primary components: (1) Learning the hidden space of user texts by two neural networks; (2) word generation and mass inference; (3) mass combination and keyword extraction. It consists of three primary phases: (1) learning the hidden space of user texts; (2) word generation and mass inference; and (3) mass combination and keyword extraction. In the first phase, user texts are grouped into small batches according to timestamps. Each batch is used for separately training two types of neural networks, the Generative Adversarial Network (GAN) and the Variational Autoencoder (VAE). In the second phase, the generators in the trained VAE and GAN work independently as two *experts* to generate bunches of tokens for modeling user preferences. Each bunch is considered as one piece of evidence, and is transformed into the so-called mass function in evidence theory by maximum a posterior estimation. In the final phase, Dempster's rule is utilized for combining the two independent pieces of evidence into an overall mass function. This mass is used for keyword extraction to form the user preferences within a specific time span. The experiments on short text data sets are conducted to inspect the efficiency of the proposed model on various evaluation metrics. Additionally, the output profile is also used for visualization, which is useful in many practical applications (Section 4.7.4). Figure

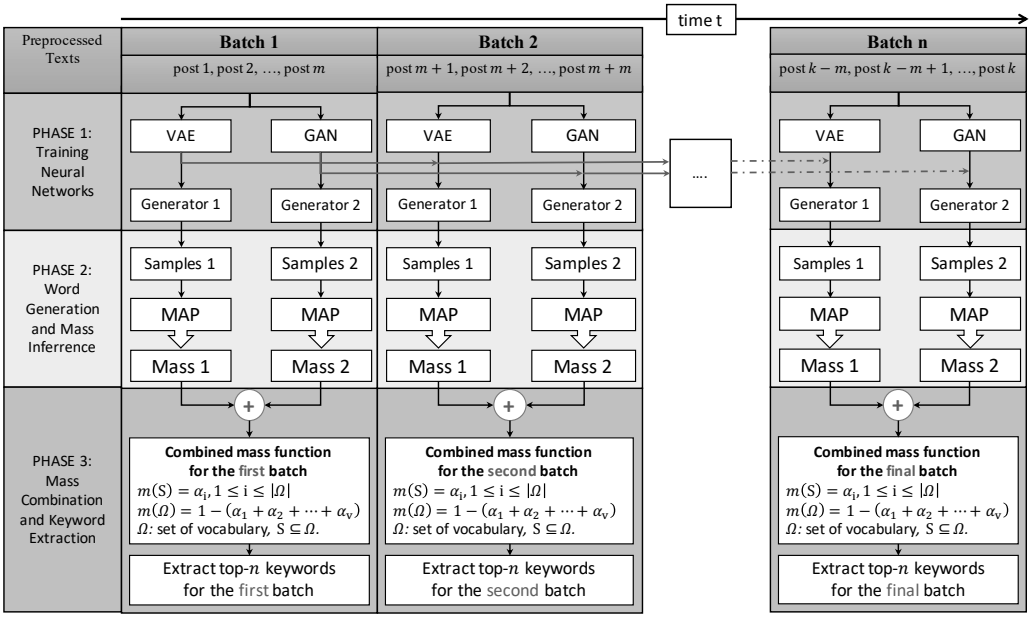


Figure 4.1: The proposed framework for capturing the dynamics of user preferences using short texts.

4.1 provides an overview of the proposed framework. The following sections will describe these components in details.

4.2 Learning the hidden space of user texts

User texts are grouped into small batches, each of which is used for learning the impact information which is useful for the generating step. Particularly, two deep neural networks, the Generative Adversarial Network (GAN) [10] and the Variational Autoencoder (VAE) [11], are trained separately by feeding the embedding vector of words extracted from the current batch. In this work, we used pre-trained word vectors from Glove model which was proposed by Pennington et al. [99]. Particularly, the word vector of each keyword is extracted thanks to the Glove model. Then these vectors serve as input data for training the two networks, VAE and GAN. The goal of this training process is to simultaneously satisfy these criteria: (1) input tokens could be reconstructed with highly similar scores; (2) the training error is minimum; (3) general information of user texts is kept.

The optimization functions we used for training the VAE and GAN are

similar to its original forms as shown in (4.1) and (4.2), respectively.

$$\mathcal{L}_{vae}(\theta,\phi) = -\sum_{j=1}^{J}\frac{1}{2}\left[1+\log\sigma_i^2-\sigma_i^2-\mu_i^2\right] + \frac{1}{L}\sum_{l}\log p\left(\vec{x_i}|\vec{z}^{(i,l)}\right) \tag{4.1}$$

where J is the number of dimensions in the Gaussian latent $\vec{z} \sim \mathcal{N}(\vec{\sigma}, \vec{\mu})$, $q_\theta\left(\vec{z}|\vec{x_i}\right)$ is the approximate posterior, L is the number of samples stochastically drawn regarding to reparametrization trick, and (θ, ϕ) are the parameters of the network to be optimized.

$$\min_G \max_D \mathcal{L}_{gan}(D,G) = \mathbb{E}_{\vec{x}\sim p_{data}(\vec{x})}\left[\log(D(\vec{x}))\right] + \mathbb{E}_{\vec{z}\sim p(\vec{z})}\left[\log\left(1-D(G(\vec{z}))\right)\right] \tag{4.2}$$

where G is a differentiable function defined by a multilayer perceptron with parameters θ_g, known as the generator, and D is a second multilayer perceptron $D(\vec{x};\theta_d)$ that outputs a single scalar (0 or 1), known as the discriminator. $D(\vec{X})$ represents the probability that $\vec{x}$ came from the data rather than p_g.

The architectures of VAE and GAN are hyperparameters depending on the dimension d of word embedding vectors (e.g., 25, 50, 100, or 200). Besides, data augmentation is also applied for increasing the number of input instances before being fed into the training processes. At this step, we randomly select some components in the embedding vectors, then adjust these components (adding or subtracting randomly) by a predefined epsilon ϵ (e.g., $\epsilon \in [10^{-1}, 10^{-6}]$). To ensure that this technique does not dramatically change the characteristics of input data, we just keep augmented tokens that are highly similar with original ones by considering cosine similarity (e.g., *cosine value* ≥ 0.85).

4.3 Word generation

After being trained, the latent spaces in VAE and GAN are used as two independent generators in generating words that represent user preferences at the current batch. Each generator (VAE's and GAN's) is formed by two sources. The first source is trained by texts in the current batch. The second source comes from VAE's and GAN's in *all* previous batches with a discounted weight determined by equation (4.3):

$$G_{vae|gan}^{(t)} = \begin{cases} G_{vae|gan}^{batch_t} & \text{if } t=1 \\ \alpha * \sum_{i=1}^{t}(1-\alpha)^{t-i} G_{vae|gan}^{batch_i} & \text{if } t>1 \end{cases} \tag{4.3}$$

where $G_{vae|gan}^{batch_t}$ is the generator of VAE model or GAN model learned from the batch t^{th} of user texts. The discounted weight α is bounded between 0 and 1. It reflects how much the $G_{vae|gan}^{(t)}$ relies on the information extracted from texts in the current batch rather than the information extracted from historical texts in previous batches. By this way, our approach uses both information extracted from the current batch and *all* pieces of information extracted from all historical batches for the generating task. This task is described in Algorithm 5. In that algorithm, the weight α, the lower bound ϵ_1, and the upper bound ϵ_2 are hyper-parameters which values could be tuned by using the testing set when building the model. The English vocabulary could be employed from common built-in libraries such as Wordlist corpora in the NLTK python library. Additionally, we integrated two concepts in Demster-Shafer theory into this algorithm to cope with special cases, the total ignorance (denoted as the set Ω) and the open-world (denoted as the set $\emptyset$). The empty set $\emptyset$ copes with unknown tokens in user vocabulary. When conditions of unknown tokens are satisfied, the model will generate a *new* word in English instead of user vocabulary. This approach makes our model flexible in predicting what *new topics* users may concern in future. This is essential for capturing the dynamic change of user preferences over time. The set Ω eliminates too common words (not stop words) like thing(s), object(s), or people because these words are similar to many other words. This is not useful for our prediction task.

4.4 Mass functions: inference & combination

The generating process in previous step results in two bunches of keywords, the first bunch is created by the latent space in VAE and the second bunch is made by GAN's generator. These two bunches are then used for quantitatively transforming into the mass functions corresponding to VAE and GAN, respectively. Then, two subtasks need to be solved are: (1) inferring mass functions; (2) combining two mass functions into an overall mass that represents user preferences at time interval t. The first task can be solved by maximum a posterior estimation. The second task's solution could be found by utilizing the Dempster's rule. Details are given as follows:

Inferring Mass Functions

Let $\mathcal{W} = \{w_1, w_2, ..., w_N\}$ be a set of N independent, identically distributed draws from a multinomial distribution $\mathcal{V}$ of size V. In this case, each element in $\mathcal{W}$ is a combinatorial extension of one bunch of words that are most

Algorithm 5: DST-based Word Generation

Input: current VAE/GAN, previous VAE/GAN, lowerBound ϵ_1, upperBound ϵ_2, User Vocabulary, English, and number of draws N

Output: A bunch of generated word vectors that represent user preferences

Compute generator at the current timestamp $G^{(t)}$ via (4.3)
Let set $S = \emptyset$
Let the count for Ω set $count_\Omega = 0$
while $|S| \leq N$ **do**
    Generate one word vector v by $G^{(t)}$
    Extract top-n keywords in user vocabulary that are most similar to v, denoted as $key_{largest}, ..., key_{smallest}$
    if $sim(key_{largest}, v) \leq \epsilon_1$ **then**
        Extract top-k keywords in English that are most similar to v
        Add top-k keywords into S
        Extend User Vocabulary with these new tokens (a set of new words is added into S)
    else if $sim(key_{smallest}, v) \geq \epsilon_2$ **then**
        $count_\Omega$ += 1
        (Nothing is added into S because of drawing a common word)
    else
        Add top-n keywords to S
Return S, $count_\Omega$

similar to the word vector generated by the generating process. The set $\mathcal{V} = \{set_1, set_2, ..., set_{m=2^V}\}$ is the power set of the set of unique tokens built from the current batch in user corpus. This vocabulary includes *fresh* tokens created by Algorithm 5. For instance, if $\text{Vocabulary}_t = \{a, b, c\}$, then $\mathcal{V} = \big\{\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\big\}$. For each draw, the likelihood of an element in $\mathcal{V}$ is computed by (4.4).

$$L\big(\vec{p}|\vec{w}\big) = p\big(\mathcal{W}|\vec{p}\big) = \prod_{i=1}^{N}\prod_{j=1}^{V} p_t^{[w_i = set_j]} = \prod_{j=1}^{V} p_t^{n_j} \tag{4.4}$$

$$\sum_{j=1}^{V} n_j = N \text{ and } \sum_{j=1}^{V} p_j = 1 \tag{4.5}$$

where n_j is the total number of times the set set_j is observed in $\mathcal{W}$. Each set in the power set $\mathcal{V}$ is assumed to follow a multinomial distribution, denoted as $Mult(set_i \in \mathcal{V}|\vec{p})$, where $\vec{p}$ is the probability that the set set_i is observed when drawing. Applying Bayes' theorem to infer the posterior distribution yields,

$$p\big(\vec{p}|\mathcal{W}, \vec{\alpha}\big) = \frac{\prod_{n=1}^{N} p(set_n|\vec{p})p(\vec{p}|\vec{\alpha})}{\int_{\mathcal{P}} \prod_{n=1}^{N} p(set_n|\vec{p})p(\vec{p}|\vec{\alpha})\, d\vec{p}} \tag{4.6}$$

To make the computational process feasible, Dirichlet distribution on the prior parameter $\vec{p}$ is applied on the likelihood to form conjugate distribution pairs as shown in (4.7).

$$\vec{p} \sim Dir(\vec{p}|\vec{\alpha}) = \frac{\Gamma\Big(\sum_j^V \alpha_j\Big)}{\prod_{j=1}^{V}\Gamma(\alpha_j)} \prod_{j=1}^{V} p_j^{\alpha_j - 1} \tag{4.7}$$

where $\vec{\alpha} = [\alpha_1, \alpha_2, ..., \alpha_j, ...]$ is the concentration parameter vector, each component α_j corresponds to p_j in $\vec{p}$. Because the denominator of (4.6) is a normalization factor, **maximum a posterior** estimation the right hand side of this equation means solving the constraint optimization problem defined by (4.8)

$$\underset{\vec{p}}{\text{argmax}} \quad \prod_{j=1}^{V} p_j^{n_j} \times \frac{\Gamma\Big(\sum_j^V \alpha_j\Big)}{\prod_{j=1}^{V}\Gamma(\alpha_j)} \prod_{j=1}^{V} p_j^{\alpha_j - 1} \tag{4.8a}$$

$$= \underset{\vec{p}}{\text{argmax}} \quad \prod_{j=1}^{V} p_j^{n_j + \alpha_j - 1} \times \frac{\Gamma\Big(\sum_j^V \alpha_j\Big)}{\prod_{j=1}^{V}\Gamma(\alpha_j)} \tag{4.8b}$$

$$\text{subject to} \quad \sum_{j=1}^{V} p_j = 1 \tag{4.8c}$$

Apply Lagrange multiplier method for solving this constraint optimization problem, we obtain the solution in (4.9)

$$p_j = \frac{n_j + \alpha_j - 1}{\sum_{j'=1}^{V} (n_{j'} + \alpha_{j'} - 1)}, \forall j \in [1, V] \tag{4.9}$$

By applying (4.9), mass functions associated with each bunch of samples drawn by VAE's generator and GAN's generator are determined as in (4.10), (4.11), respectively.

$$m\big(S_i\big) = \frac{(\#\text{times the set } S_i \text{ appears in } \mathcal{W}) + \alpha_i - 1}{|\mathcal{W}| + \sum_{i=1}^{V} \alpha_i - V}, \tag{4.10}$$

$$m(\Omega) = \frac{(\#\text{times } \Omega \text{ appears in } \mathcal{W}) + \alpha_\Omega - 1}{|\mathcal{W}| + \sum_{i=1}^{V} \alpha_i - V}. \tag{4.11}$$

We verified that the sum of all masses is equal to 1 $\big($i.e., $m(S_i) + m(\Omega) = 1, \forall i \in \big[1, V\big]\big)$. In the next step, Dempster's rule is used to combined two pieces of evidence into an overall mass function which is the representation for user preferences within the time interval t.

Combining masses from multiple sources

Suppose that we have a set $\mathcal{M} = \{m_1, m_2, ..., m_M\}$ where $|\mathcal{M}| = M$, each m_i is a mass function corresponding to the i^{th} source over a frame of discernment Ω. $\Omega = \big\{S_1, S_2, ..., S_V\big\}$ is a set of all possible subsets S_i in $\mathcal{V}$, $|\Omega| = V$. In our case, $M = 2$, and each mass m_i is induced from one bunch of tokens generated by either VAE or GAN via (4.10), (4.11) in the previous step. Here we consider each of the predictions made by VAE and GAN as an evidential source that is valuable for reasoning process. Combining these two sources gives the results determined by the equations (4.12) and (4.13).

$$m_1 \oplus m_2\big(S_i\big) = \frac{\sum_{S_i' \cap S_i'' = S_i} m_1(S_i') \times m_2(S_i'')}{1 - \kappa}, \tag{4.12}$$

$$m(\Omega) = \frac{\big[m_1(\Omega) \times m_2(\Omega)\big]}{1 - \kappa} \tag{4.13}$$

where κ is the degree of conflict between two sources identified by (4.14):

$$\kappa = \sum_{A' \cap A'' = \emptyset} m_1(A') \times m_2(A'') \tag{4.14}$$

4.5 Profile extraction

In this phase, basing on the overall mass function in the previous step, the so-called pignistic probability principle defined in (2.10) is used for computing the weight of all singletons in Ω. Then, top-n concepts with highest probability are extracted to define the user profile at the time interval t via (2.11). The entire process of the proposed framework is represented in **Algorithm 6**.

Algorithm 6: The entire process for finding user preferences within a specific time interval

Input: a positive integer n, $batch_t$ of user texts, the generator $G^{(t-1)}$ at the previous time span
Output: top-k keywords

Phase 1:
  Learn the latent space of current $batch_t$ via VAE
  Learn the latent space of current $batch_t$ via GAN
  Calculate the generator $G_{vae}^{(t)}$ via (4.3)
  Calculate the generator $G_{gan}^{(t)}$ via (4.3)
Phase 2:
  Generate the first bunch of words via $G_{vae}^{(t)}$ and Alg. 5
  Generate the second bunch of words via $G_{gan}^{(t)}$ and Alg. 5
  Infer the mass function corresponding to $G_{vae}^{(t)}$ via (4.10), (4.11)
  Infer the mass function corresponding to $G_{gan}^{(t)}$ via (4.10), (4.11)
Phase 3:
  Combine two mass functions via (4.12), (4.13)
  Compute pignistic probability of singletons via (2.10)
  Sort pignistic probabilities in descending order
  Pick up top-n keywords called set S
Return S as user preferences at time t
End Algorithm.

4.6 The rationality of the proposed approach

Our model just uses a few of texts in user corpus for training VAE and GAN separately. After being trained, the generators of VAE and GAN work independently as two "experts" in predicting user preferences within a specific

time interval. Each prediction made by one expert is viewed as *a piece of evidence* in Dempster-Shafer theory. After that, we quantitatively fuse these two pieces of evidences into an overall mass function. This mass function is the basis for extracting top-n keywords to define the user profile at time t. By this way the proposed framework is able to work properly when we insert more predictors to work as a new "expert". Additionally, when generating words at time t, the model uses two sources of information. The first source is extracted from texts in the current batch. The second source comes from the *representative* information learned from texts in *all* previous batches with a discount weight. This way ensures that the proposed framework uses both current texts and historical texts to predict the dynamics of user preferences at time t. Additionally, the Algorithm 5 is able to generate *fresh* tokens that may appear *anywhere* in user corpus. Therefore, this approach is reasonable in practice because user concerns on various topics usually change over time, especially customers on e-commerce systems.

4.7 Experimental results

This section introduces the data sets, experiments, and evaluation metrics for comparing the efficiency between the proposed framework and baseline models. Finally, we conduct an extra experiment which uses the output of the proposed framework to visualize the dynamics of user preferences over time.

4.7.1 Data sets and baseline models

The experiments are conducted ontwo data sets: (1) Twitter data set - a publicly available data set collected from Twitter [5][1]; (2) Facebook data set which is crawled by our research team. Twitter data set consists of **1189** users after being preprocessed. There were around **2689** tweets per user, each of which contains about **9** words on average. Facebook data set consists of **500** users after being preprocessed. There were around **816** posts per user, each of which contains about **36** words on average. All texts in both data sets are passed through a preprocessed pipeline, including noise removal, tokenization, and normalization. Particularly, we carried out the following tasks sequentially: eliminate HTML tags and white spaces, delete special characters, lowercase all texts, remove all stop words, apply stemming process to transform a word into its base form (e.g., "loving" or "loves" to "love", "wolves" to "wolf").

[1]Available at https://bitbucket.org/sliang1/uct-dataset/get/UCT-Dataset.zip

We conducted experiments to make comparisons between our proposed method and the baseline models that work efficiently on text summarizing and keyword extraction as described as follows [109]: Baseline models used in the experiments are described as below:

- **GSDMM**: this is a Dirichlet Multinomial Mixture model with a collapsed Gibbs Sampling technique for clustering short texts. The model is proposed by Yin *et al.* in 2014 [104].
- **Rake**: the Rapid Automatic Keyword Extraction model proposed by Rose et al. in 2010 for text description [105].
- **TextRank**: this is a graph-based model for summarizing text proposed in 2016 [106, 107].
- **TFIDF**: this is a traditionally statistical model that estimate how important a keyword is to a document in a corpus[2] [108].
- **LDA**: the Latent Dirichlet Allocation model uses a Gibb Samping technique to topic modelling. We based on this model to derive the algorithm for user profiling problem as shown in **Algorithm 4**.
- **YAKE**: an unsupervised, statistical model proposed in 2020 for keyword extraction from single documents using multiple local features [110];
- **PositionRank**: An unsupervised, graph-based model proposed in 2017 for keyphrase extraction from documents [111].
- **MultipartiteRank**: an unsupervised model proposed in 2018 for keyphrase extraction with multipartite graphs [112]
- **dst**: the proposed model bases on Dempster-Shafer theory and deep generative networks.

The procedures of LDA, GSDMM, YAKE, PositionRank, MultipartiteRank, RAKE, TextRank, and TFIDF models are similar to the models we presented in Section 3.7.2.

[2]https://en.wikipedia.org/wiki/Tf–idf

4.7.2 Evaluation metrics

The experiments are to answer these criteria: **CR1.** How is the overall performance of all models in predicting the dynamics of user preferences using short texts according to common evaluation metrics? **CR2.** How is the overall performance of all models in capturing the abstract concepts shared between words in user texts that implicitly reflect their preferences at time span t? **CR3.** How does the size of word vectors impact the efficiency of the proposal framework? **CR4.** How is the time complexity of all models when working on practical data sets? and **CR5.** Is there an appropriate approach for visualizing the dynamics of user preferences over time?

Common metrics such as standard precision at n, Mean Reciprocal Rank (MRR), and Mean Average Precision (MAP) are usually used for evaluating **CR1** [113]. The semantic precision, denoted as s-precision, could be used for evaluating **CR2** [4]. Executive time is usually used for answering **CR3** in practice. We record the semantic precision performance of all models at different sizes of word vectors for answering **CR4**. Finally, the approach for the **CR5** could be found by using the output of our proposed framework to visualize the changes of user preferences on five given topics over time.

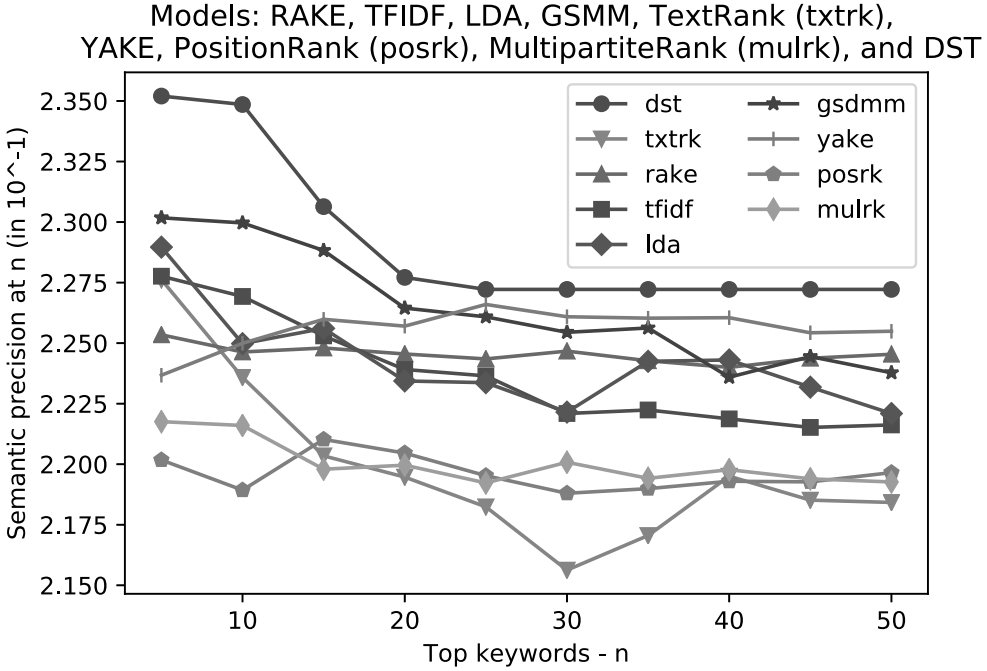


Figure 4.2: Twitter dataset: Semantic precision.

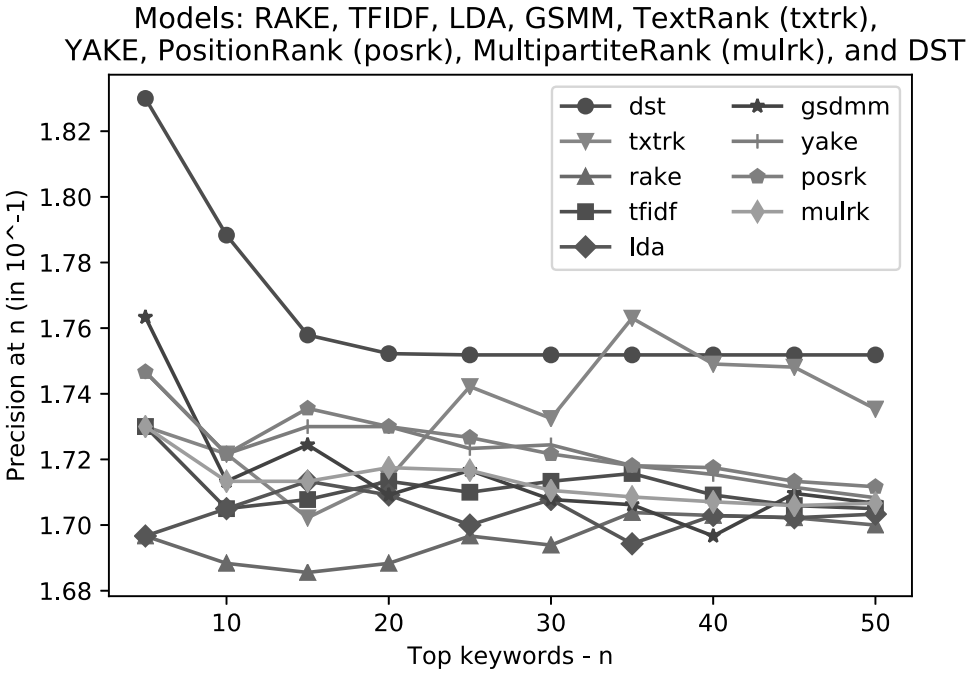


Figure 4.3: Twitter dataset: Standard precision.

4.7.3 Results

Each of the data sets is divided into two subsets, the train set and the test set, with the ratio 8/2. The train set is used for tuning hyper-parameters such as network architectures, learning rate, and slope value in activation functions. Additionally, for each user corpora, we further group preprocessed texts into $n = 12$ batches corresponding to 12 time intervals. Each batch consists of around 134 posts, 95 percentages of posts in the batch are used for training and inferring. In this stage, we also applied data augmentation with an amount of 20 folds to increase the size of input data. Five remaining percentages are used as candidate keywords (eliminated stop words) which are sorted in an ascending order of time for evaluation purpose. At each batch of user texts, the number of keywords extracted by all competitors varies from 5 to 50 with step size 5 (e.g., $[5, 10, ..., 50]$). We report the average result of 30 runs in the test set.

Figure 4.2 to Figure 4.15 show the performance of textrank, rake, tfidf, lds, gsdmm, and our proposed method (dst) on following metrics: semantic precision at n (the number of keywords in prediction), standard precision at n, MRR at n, MAP at n, and runtime at n. As illustrated in Figure 4.2 and Figure 4.4, dst model outperforms all the baselines in almost values of n

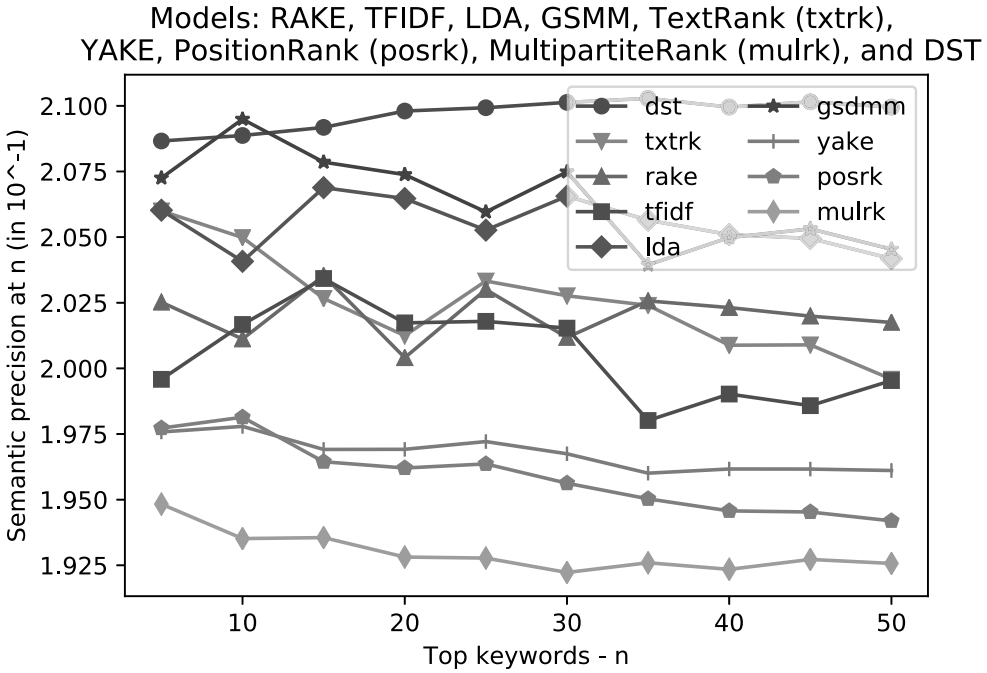


Figure 4.4: Facebook dataset: Semantic precision.

according to semantic precision. For the performance on standard precision metric, the fluctuation of dst model is more stable when the number of keywords increases. And in many values of n, our proposed model outperforms the baselines, especially at length $35, 40, 45$, and 50.

In MRR and MAP metrics, the dst model is more stable in comparison with the baselines when making first correct prediction (MRR) and when ranking a list of keywords to be appeared for representing user preferences in the next time step (MAP). This is because the ranking made by Smets principle in Dempster-Shafer theory does not depend on the number of keywords to be extracted. Overall, these results validate the efficiency of our proposed framework for the first two research questions **C1** and **C2**.

Turning to **C3**, we estimated the semantic precision of all models by varying the size of word embedding vectors within the range $[25, 50, 100, 200]$. For each vector size, we tuned the networks of VAE and GAN then selected the architecture which gave the highest performance. Figure 4.10 to 4.13 show experimental results on Twitter data set. We observe that: (1) the word vector of size 50 give the best performance compared to the remaining sizes; and (2) the ranking of all models is nearly identical, and is distributed into three categories with the descending order as follow: *dst* >

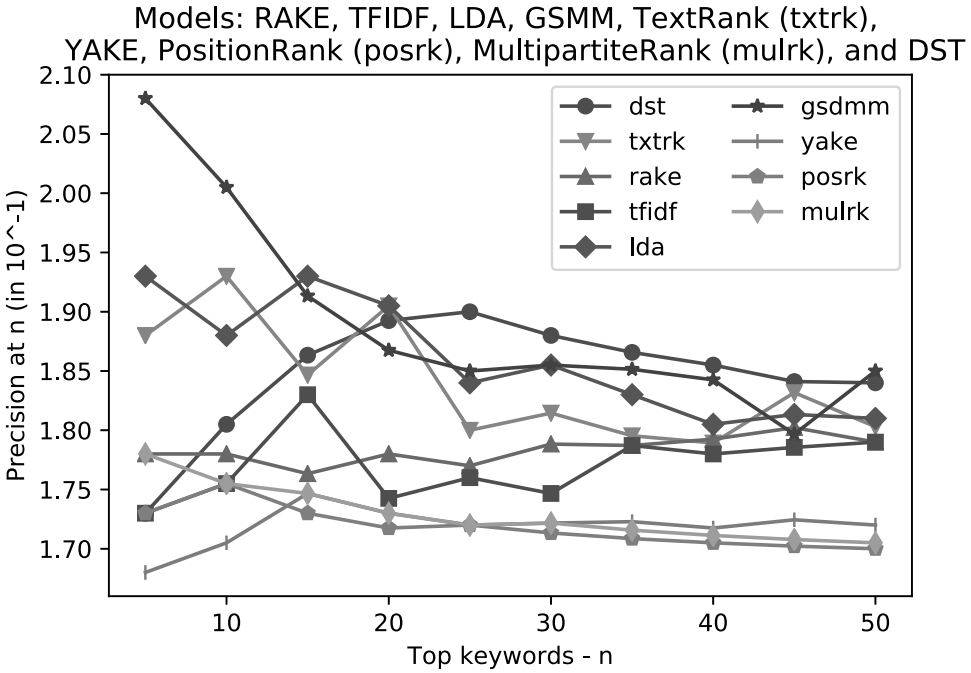


Figure 4.5: Facebook dataset: Standard precision.

$(lda, gsdmm, yake, tfidf, yake) > (posrk, mulrk, txtrk)$. This observations again validate the efficiency and the stability of the proposed framework when the size of word vector is dramatically changed. Additionally, the result of this experiment type also reveals an appropriate approach for visualizing the dynamics of user preferences over time as proposed in Section 4.7.4.
Turning to **C4**, the dst model gives the second worst performance on runtime when being compared with baselines. This is one of the disadvantages of our proposed framework because Dempster-Shafer theory considers all subsets of a given set when inferring and combining mass functions. In theory, the time complexity of our proposal is proportional to exponential functions in worst case.

4.7.4 User preference visualization

From semantic precision we propose an approach for visualizing the change of the level user concern on common topics over time. The approach is briefly described as follows: Predefined keywords in five common fields, including Education, Politics, Science & Technology, Entertainments, and Economics,

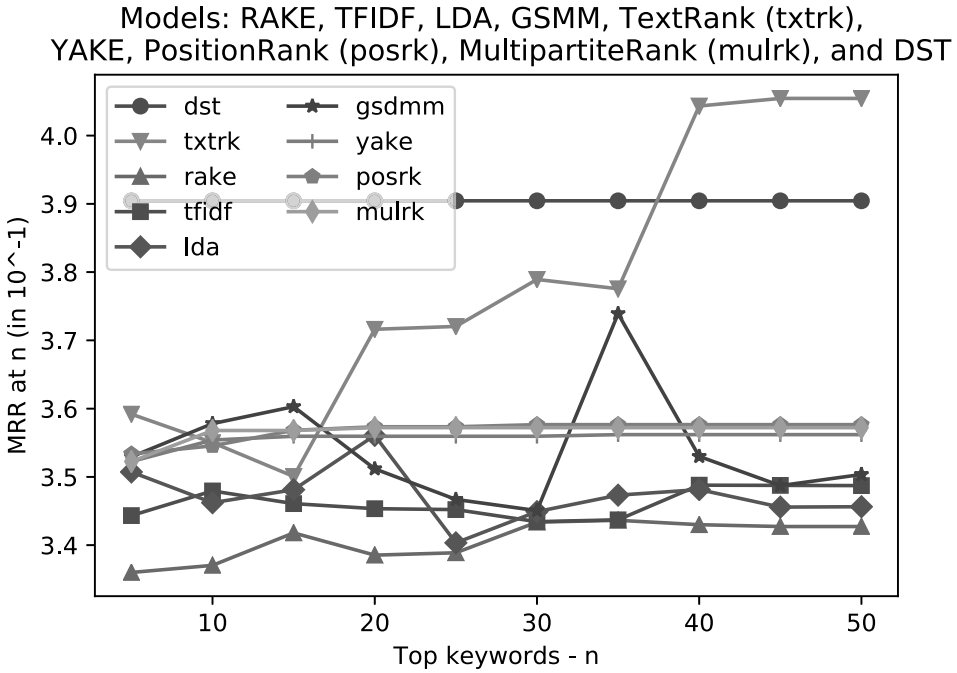


Figure 4.6: Twitter dataset: MRR.

are listed by experts[3]. The semantic similarity between each of predicted words at time t and all words in the predefined keywords of the given topic is calculated. Then average value is computed and plotted in a graph to represent the preference level of the user on that topic at time t. The entire procedure is provided in **Algorthm** 7. Figure 4.16 shows the visualization of two randomly selected users. We observed that the most concerned topic of the user 1 is Education while the most concerned topic of the user 2 is Entertainments. Additionally, the level of concern on Politics is on the upward trend for the user 1. This observation may reveal the insight that the user 1 is more concerned in Politics over time. Besides, the results in this visualization approach are helpful and interpretable because cosine similarity is bounded in $[-1, 1]$. This value can be interpreted as "dislike" (-1), "neutral" (0), or "like" (1). Many applications such as recommendation systems can utilize these insights to offer appropriate items that best match user demand. Another application is that users can be clustered into groups regarding to their preferences so that a number of useful insights may be revealed for decision making in various business contexts.

[3]There are 6 annotators who are experts in the field participating in our experiments.

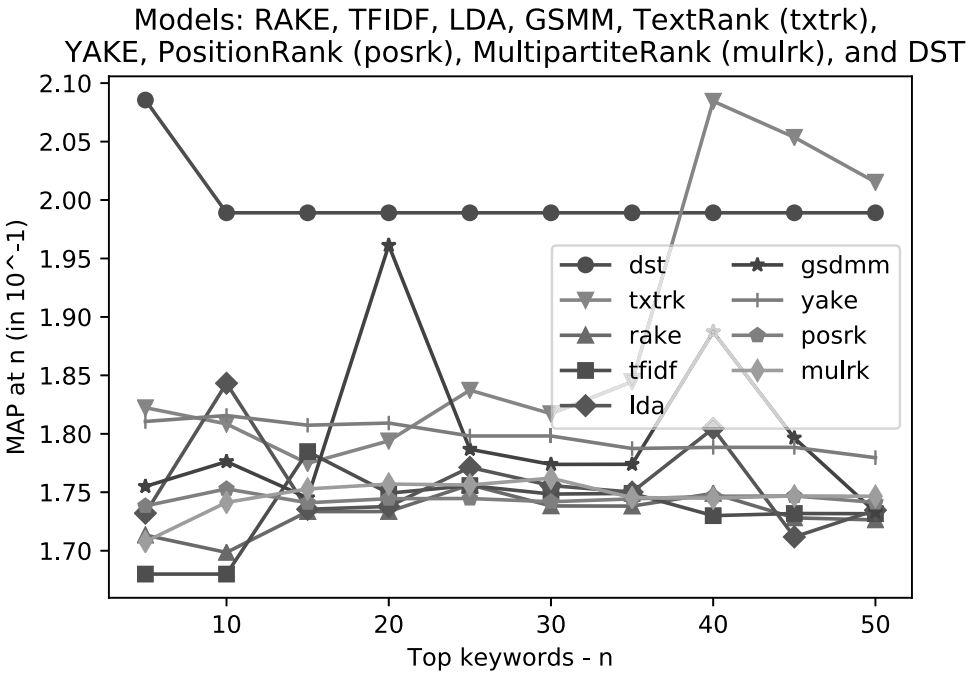


Figure 4.7: Twitter dataset: MAP.

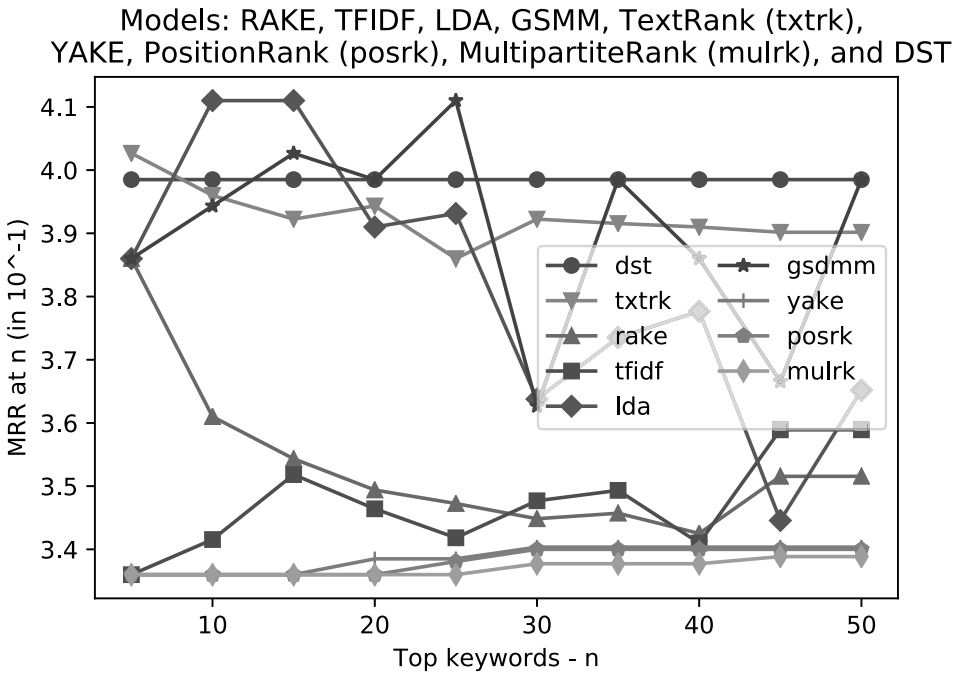


Figure 4.8: Facebook dataset: MRR precision at k.

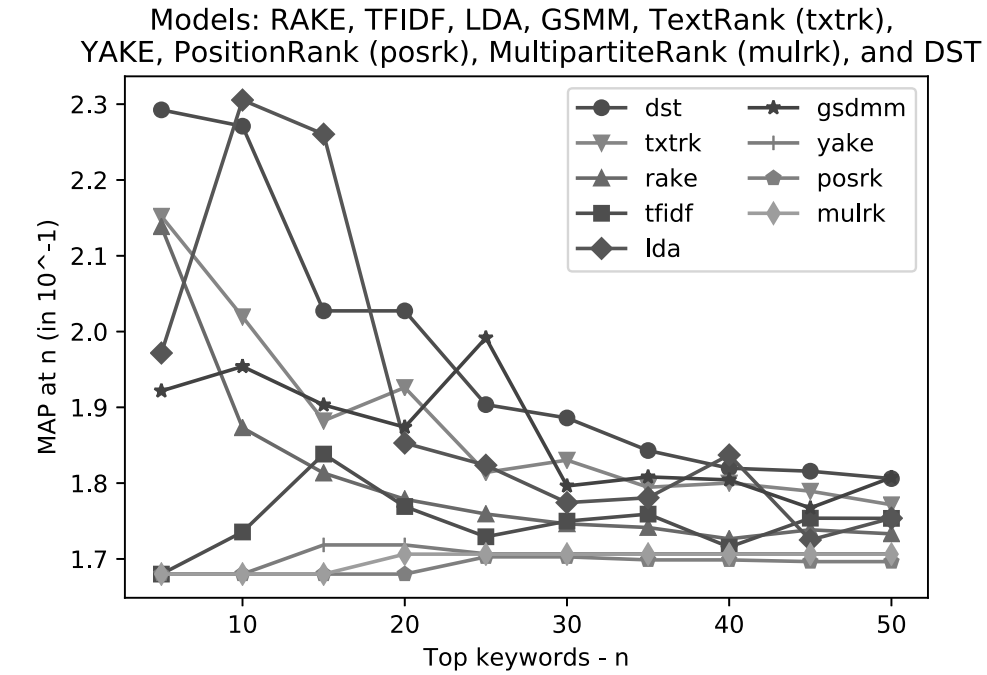


Figure 4.9: Facebook dataset: MAP precision at k.

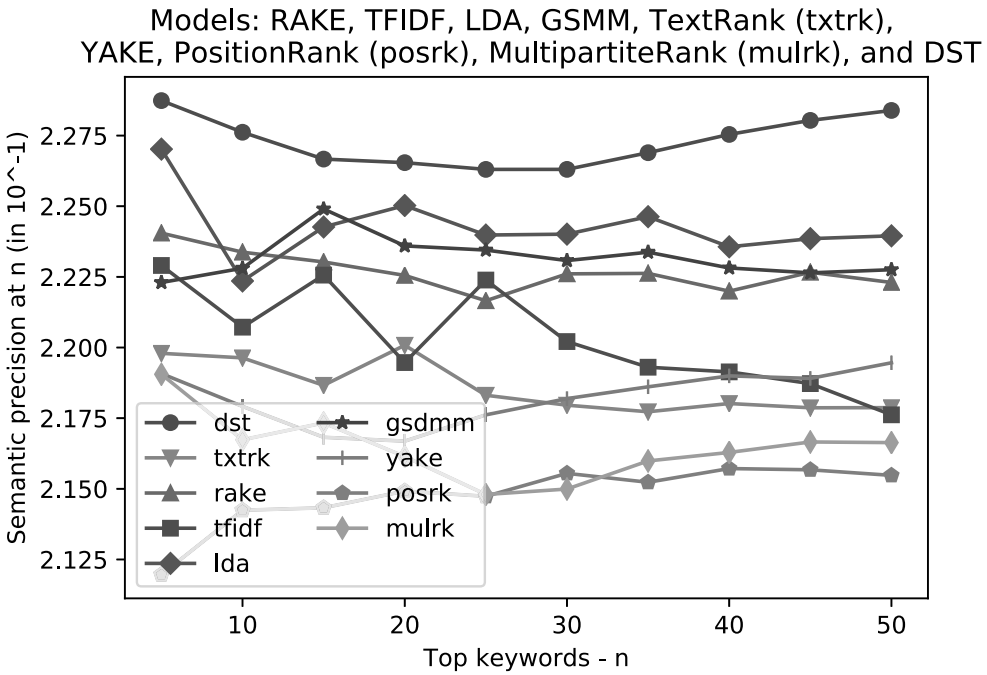


Figure 4.10: Twitter dataset: embedding vectors' size = 25.

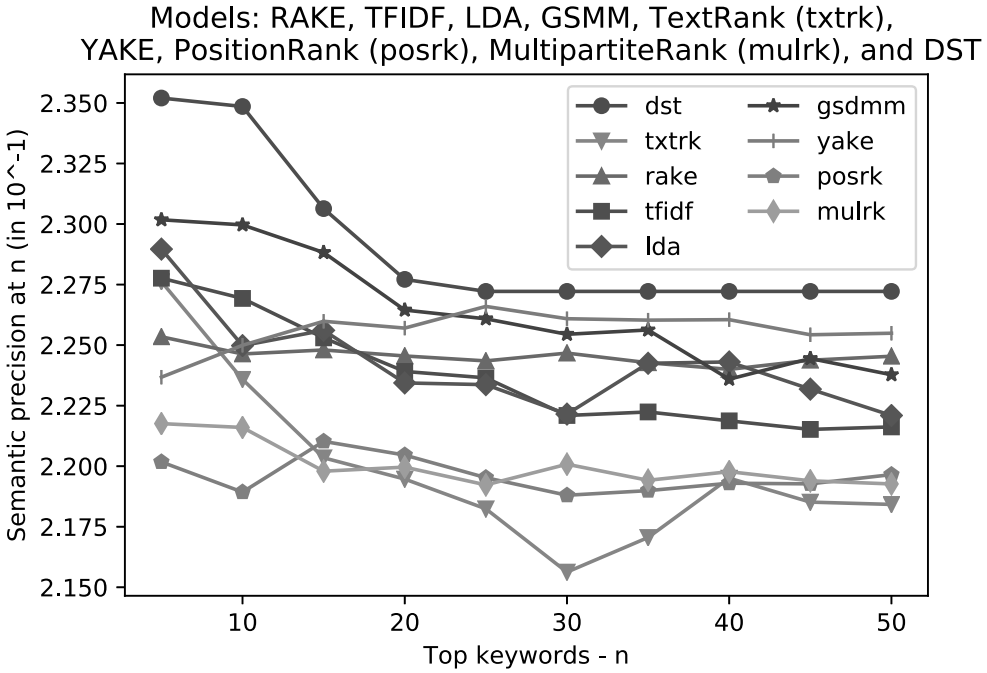


Figure 4.11: Twitter dataset: embedding vectors' size = 50.

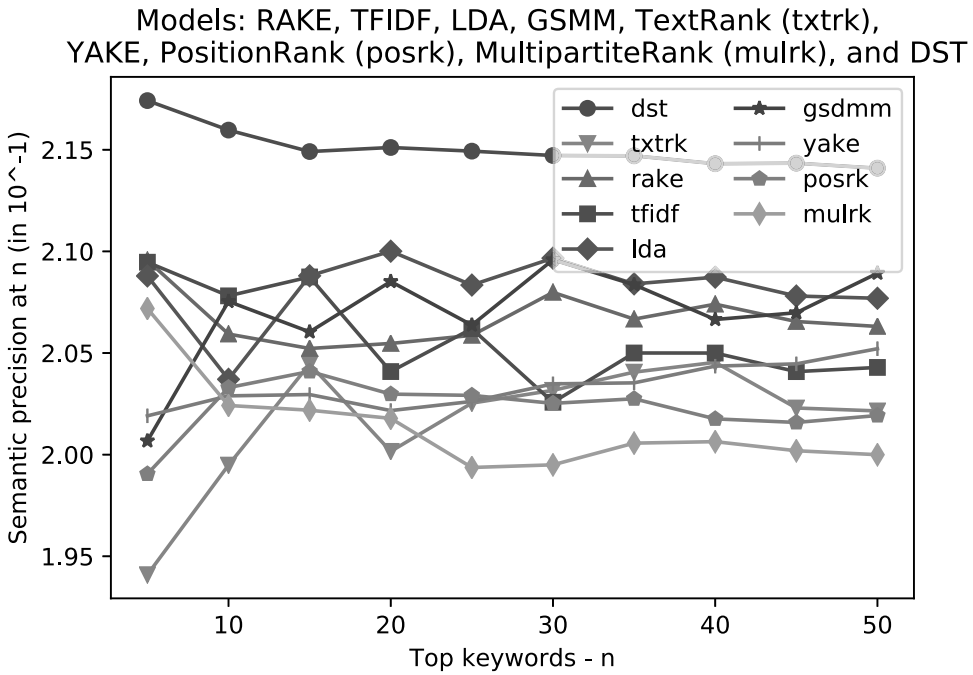


Figure 4.12: Twitter dataset: embedding vectors' size = 100.

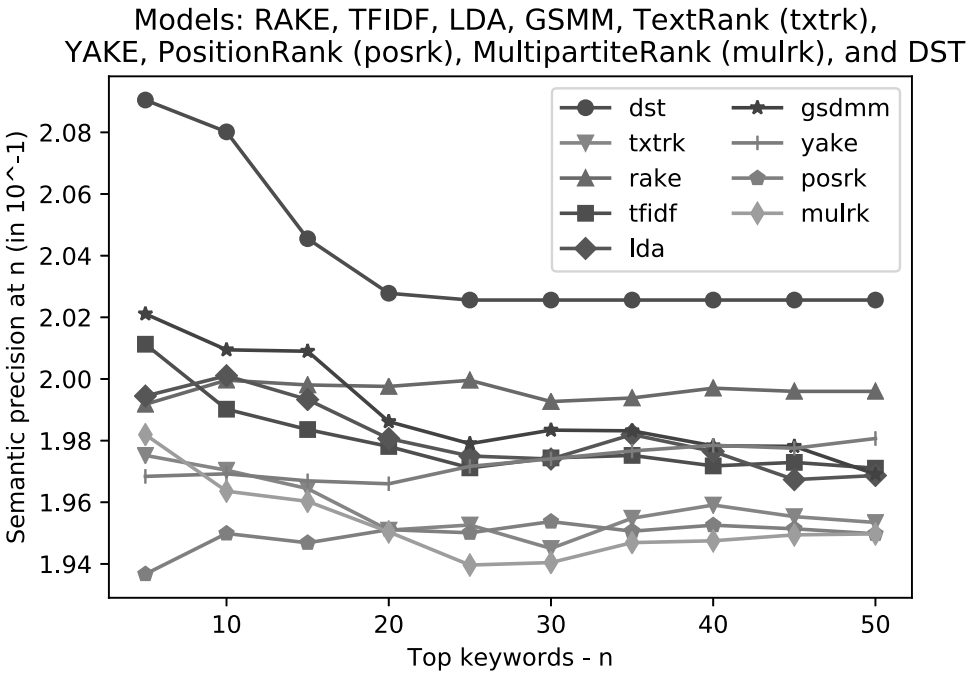


Figure 4.13: Twitter dataset: embedding vectors' size = 200.

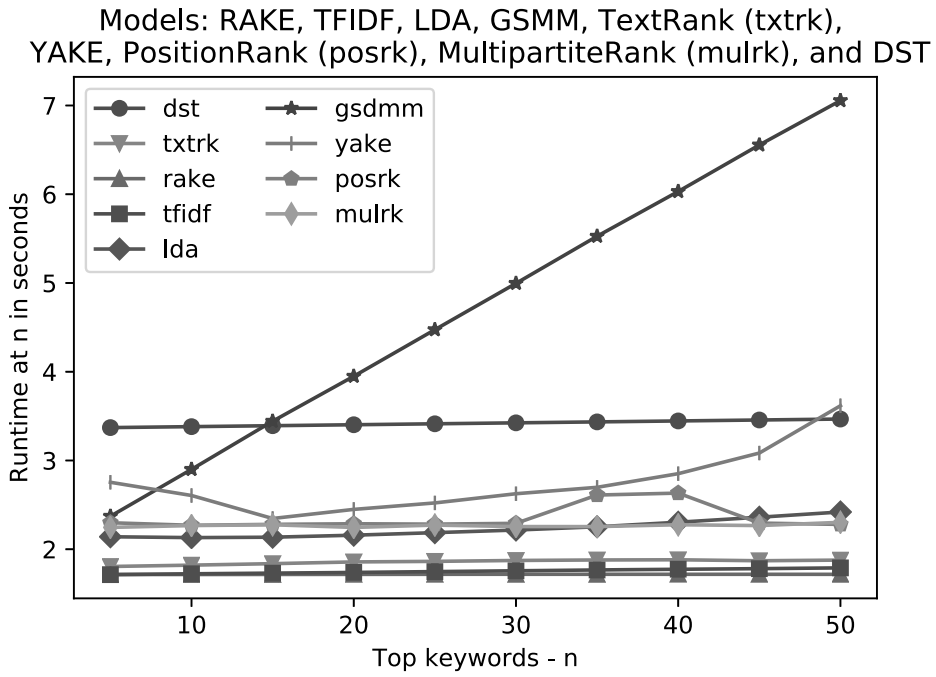


Figure 4.14: Twitter dataset: Runtime.

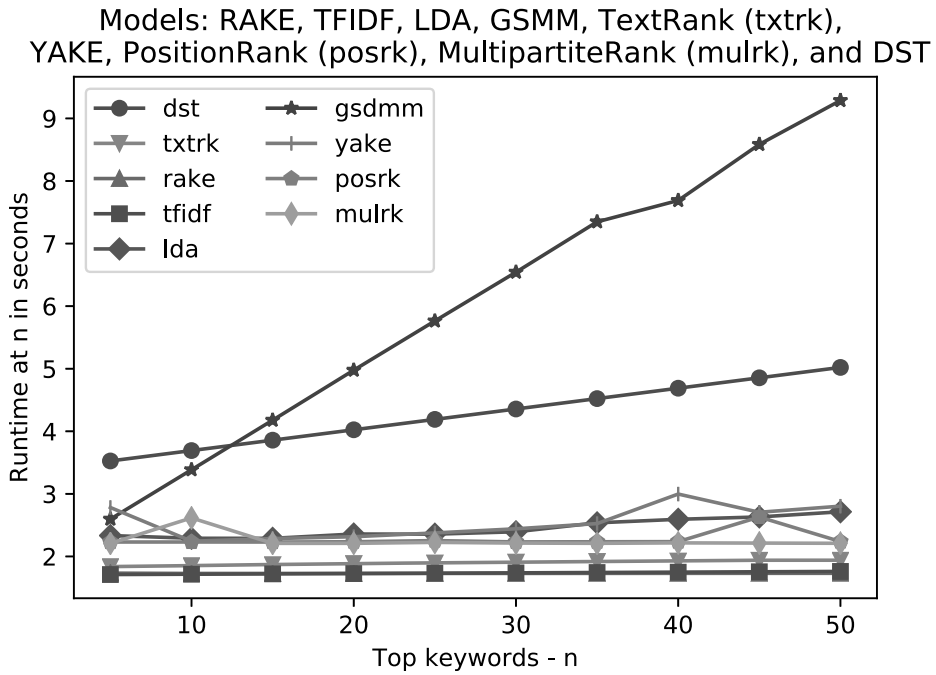


Figure 4.15: Facebook dataset: Runtime.

Algorithm 7: Estimation of the level of user concern on a given topic at specific time t

Input: A list of top m keywords representing topic i
Input: A list of top k keywords at time t of a given user u_i
Input: Pre-trained word vector model
Output: A real number between $[-1, 1]$ that quantitatively represents user concern on the topic i at time t

Get word vectors of all keywords in topic i, called the set T
Obtain word vectors of all keywords of user preferences at time t, called the set K
Let $S = \emptyset$
for *each word vector* w_k *in* K **do**
    for *each word vector* w_t *in* T **do**
        Compute cosine similarity $cos(w_k, w_t)$
        Add $cos(w_k, w_t)$ into the set S
Compute the average of all values in S, called r
Return r
End Algorithm

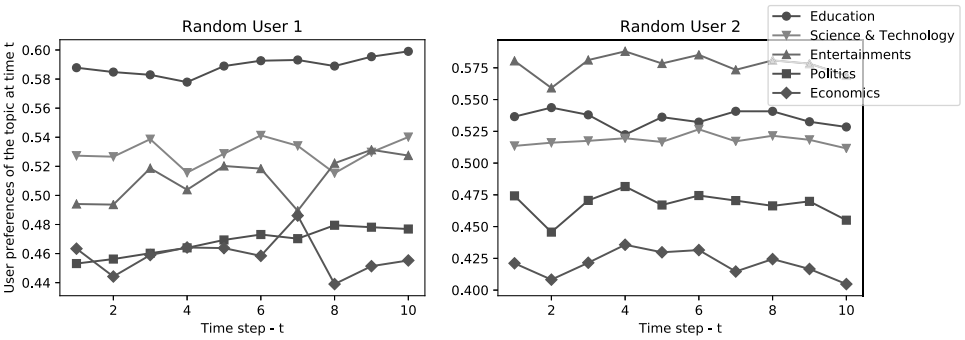


Figure 4.16: The changes of Twitter user preferences over time.

Chapter 5

Conclusion

5.1 Summary

This research integrated advancements of machine learning and evidential reasoning in Dempster-Shafer theory for the problem of *capturing user profile using short texts on social networks*. The user profiling problem is considered under two scenarios, the static profile and the dynamic profile. This research proposed two novel frameworks, each of which corresponds to one scenario.

Intuitively, the approach in the first proposed framework is to reasonably estimate the "important" level of each abstract concepts in user vocabulary. Basing on this important level, top-n concepts are extracted to define the user profile. Overall, these tasks are solved sequentially:

1. *mine* the hierarchy of abstract concepts that are hidden inside plain texts via k-means clustering;

2. *quantitatively* estimate one piece of information stored in each document that contributes to the desired profile. This task is carried out by maximum a posterior estimation;

3. combine mass via the Dempster's rule and extract the profile via the so-called pignistic probability principle in evidence theory.

All experiments are conducted on two short text data sets to validate the effectiveness and stability of the proposed model. The advantage of this approach is that the extracted profile is visualizable and interpretable. This ability may reveal useful insights for decision making in practice. Additionally, this proposed framework is feasible to be scaled in some extension. For example, the input data may come from multiple sources on web environment, each of which may exists in different formats (i.e., texts, photos,

reactions).

In the second proposed framework, the advancements of state-of-the-art generative neural networks and Dempster-Shafer theory of evidence are first integrated into a framework for capturing the dynamics of user preferences over time using short texts. The proposed framework has three primary components: (1) learning the latent space; (2) word generation and mass inference; and (3) mass combination and keyword extraction. We carried out experiments to answer five evaluation criteria for verifying the effectiveness and stability of our proposed framework. Besides, we also proposed an approach for visualizing the dynamics of user preferences over time on various topics of daily activities. The proposed approach can be integrated into applications to obtain insights which are helpful for decision making, especially in recommendation systems.

5.2 Research impacts

Thanks to key contributions of two proposed frameworks, the expected impact of this research could be listed as follows:

- This research opens a new direction for the user profiling problem in both static and dynamic contexts. In this direction machine learning techniques can be used for learning and evidential reasoning can be used for fusing and inferring the user profile. Besides, input data may not only exist in different modes, but also come from multiple sources.

- Many practical applications could be beneficial from this research. For example, there are applications such as friendship recommendations on social networks, job offers on labor markets, item recommendations on e-commerce platforms, and user clustering for decision making in many business scenarios.

- The research output could be a clue for user preference visualization which is helpful in various business contexts. Visualization is a common requirement in practice, especially e-commerce recommendation systems.

5.3 Discussions

5.3.1 The interpretability of the extracted profiles: what insights may be obtained

The performance of extracted profile was evaluated on semantic precision metric is beneficial. Particularly, the semantic score may reveal valuable insights for decision making in many business contexts. For example, assume that the scores of a specific user on three topics, including politics, education, and technology, are -0.90, 0.02, and 0.86, respectively. Then, we can conclude that this user highly concerns on technology, totally does not like politics, and is neutral in education. These insights are useful for item recommendation systems, especially on e-commerce platforms.

5.3.2 Potential applications: user clustering

Similar users could be dynamically clustered into groups in which individual shares similar interests on a specific topic. This research issue is more challenging when input data are short texts and the problem is considered under a dynamic context [98]. Due to our approach, this problem could be beneficial from the output of the proposed framework. For example, the semantic measure between pairs of keywords in user preferences may reveal *similar* users at a specific time interval. Thanks to this insight, a recommendation system can suggest an appropriate item to not only individuals but also a group of customers in the system. Additionally, these insights can help in detecting interests of users whose profiles are totally different under a *static* context but converged at some point of time under a *dynamic* context. For example, the interested topics of a politician and an athlete may differ on overall, but they could share the same concerns on cinema, sightseeing, or travelling at a specific time span.

5.3.3 The appropriate length of user profiles

The length of user profile can take a value in the set $S = 5, 10, 15, ..., 50$. This number is usually suggested by experts, and depends on applications. For example, one application needs to estimate how much a given user concerns on three topics, including technology, education, and politics. Some experts are invited, and they suggest that the number of keywords for each topics is $3, 5, 2$, respectively. Then, the profile length should be 10 in this case.

In literature, this aspect has been research in [114]. The authors claimed that the rational profile does exist by a reasonable size controlling method. In

our work, the profile length is not profoundly considered yet, it much depends on the expert committees and toward specific domains. In the future work, further experiments need to be conducted before determining an appropriate number of keywords in extracted profiles.

5.3.4 How to quantitatively and qualitatively evaluate the generators in GAN and VAE models?

Although many metrics (e.g., average log-likelihood, Inception Score, Fréchet Inception Distance (FID) score, or visual fidelity of samples) were introduced/used to evaluate the quality and quantity of generative models such as the works in [115, 116, 117, 118], many researchers pointed that extrapolation from one criterion to another is not warranted and generative models need to be evaluated directly with respect to a downstream task they were intended for [119, 115]. In our case this downstream task is to predict texts reflecting what a given user may concern in future.

Initially, VAE and GAN are designed for image data. We need to adapt these models for textual data appropriately. Therefore, the evaluation criteria usually used in qualitative and quantitative processes also need to be reasonably adjusted before being applied for our work. Particularly, the following measures are suitable and feasible enough:

- For quantitative measures, we used: (1) Token Quality Measure to subjectively evaluate the quality of generated tokens [120, 121, 122]; (2) Reconstruction Error (or Root Mean Squared Error) to objectively measure the different between reconstructed tokens and input tokens [123, 117].

- For qualitative measures, we used: (1) Nearest Neighbors to detect over fitting. Generated samples are shown next to their nearest neighbors for checking the similarity [115]; (2) Preference Judgment [124, 125, 126] to rank the fidelity of generated tokens in term of cosine similarity (objectively) and the judgment of annotators (subjectively). Besides, mode drop and collapse [127, 128] in GAN model are also taken into consideration carefully to ensure that the trained models is diverse enough when being used.

In the framework, these measures are combined into a two-step process as follows: (1) train the VAE and GAN separately white carefully checked the loss and reconstructed errors during the training process; (2) a preference judgment process is carried out by annotators to post-check the generators of two networks.

5.4 Limitations

5.4.1 Computational efficiency

Runtime is one of the most disadvantages of our approach. This is because all subsets of a given set is considered in Dempster-Shafer theory. Consequently, the time complexity is proportional to an exponential function in worst case. However, this factor could be significantly improved if the framework is restricted to consider only a limited number of focal sets. For instance, we may constrain the focal sets to be either a singleton, a 2-element set, a 3-element set, an Ω, or the empty set $\emptyset$. In this way, the time complexity is reduced to maximum a cubic function ($O(n^3)$) in the worst case, without sacrificing too much the flexibility of the proposed framework. This solution is used to be applied in previous studies such as [33, 16].

5.4.2 Word vectors should be trained on each user corpus?

In this work, the proposed frameworks employed a pretrained model, called GloVe, to convert words into vectors. Although this model was trained on a large numbers of tokens created by Twitter users, there is a fact that "different users have different manners". Therefore, word embedding model should be trained only on user own corpus. This approach may make the framework more robust in capturing the abstract concepts hidden inside plain texts created by individual user.

5.5 Future works

5.5.1 Applicable aspects of the proposed frameworks

The proposed framework would be more appreciated if their outputs could be integrated in a specific application (e.g., item recommendation). Besides, the integration could evaluate the impact or influence of the acquired user preferences. We are going to accomplish this requirement. One more research problem need to be solved, called "User Identification" across multiple platforms [129, 130, 131, 132]. This research problem aims at connecting individuals across social media sites to integrate partial information of a user for building a better profile/preference. However, seeking an efficient solution for "User Identification" is not straightforward. It is an extremely challenging task as users provide limited or no information for matching purposes,

or the connectivity among user identities across different sites is often unavailable [129, 131]. As for future work, we plan to integrate the acquired user preferences into an application for job recommendation in the LinkedIn System. In that application, we will solve the following tasks sequentially:

> **Task 1:** Collect partial information (short texts) of users in social networks. Then we mine user concerned topics or preferences via the proposed methods in this research.
>
> **Task 2:** Solve the User Identification Problem to combine the partial information of a user across multiple platforms such as Facebook, Twitter, Instagram, and LinkedIn.
>
> **Task 3:** Recommend an appropriate job for users on LinkedIn based on their personal traits and concerned topics acquired by the **Task 1**.

5.5.2 Consideration of different formats of input data

One advantage of the proposed frameworks is that they are able to work properly when data exist in many formats (such as photos, texts, or number of reactions) or come from multiple sources (such as Twitter, Facebook, and Instagram). For example, captioning techniques could be utilized for obtaining textual data from photos and the number of reactions on them such as generating image descriptions [133] or transforming objects into words [134]. Then these obtained textual data could be fed into the framework to be treated as user texts. In these cases, we just need to add an additional model into the framework to work as a new, independent "sources" in the prediction process. The choice of added models is flexible, such as Convolutional Neural Networks to cope with photos or Recurrent Neural Networks cope with textual data when the *order* between words is important. Additionally, to carry out the evaluation process in a more appropriately convinced manner, the gold keywords should be manually prepared by several human experts. Therefore, we should invite experts to involve in this process, especially in the process of profile visualization.

Publications

Journals

[1] D.-V. Vo, J. Karnjana, V.-N. Huynh. "An Integrated Framework of Learning and Evidential Reasoning for User Profiling using Short Texts." *Information Fusion*, vol. 70, pp. 27–42, 2020. https://doi.org/10.1016/j.inffus.2020.12.004.

[2] D.-V. Vo, T.-T. Tran, K. Shirai, V.-N. Huynh, "Deep Generative Networks Coupled with Evidential Reasoning for Dynamic User Preferences Inferring Using Short Texts. *IEEE Transactions on Knowledge and Data Engineering.* [**under the** 2^{nd} **revision**]

Conferences

[3] D.-V. Vo, A. Hoang, V. -N. Huynh, "An Evidential Reasoning Framework for User Profiling Using Short Texts," The Eight International Symposium on Integrated Uncertainty in Knowledge Modelling and Decision Making (IUKM), 2020, pp. 137-150, doi: https://doi.org/10.1007/978-3-030-62509-2_12.

[4] D.-V. Vo, T.-T. Tran, V. -N. Huynh, "Dynamic User Preferences Using Short Texts," The 26^{th} Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), May 16-19, 2022, Chengdu, China. [**under review**]

References

[1] J. Tang, L. Yao, D. Zhang, and J. Zhang, "A combination approach to web user profiling," *ACM Transactions on Knowledge Discovery from Data*, vol. 5, no. 1, pp. 1–38, Dec. 2010.

[2] M. M. Waldrop, "News feature: What are the limits of deep learning?" *Proceedings of the National Academy of Sciences*, vol. 116, no. 4, pp. 1074–1077, Jan. 2019. [Online]. Available: https://doi.org/10.1073/pnas.1821594116

[3] K. Balog and M. de Rijke, "Determining expert profiles (with an application to expert finding)," in *IJCAI'07*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2007, pp. 2657–2662.

[4] S. Liang, "Dynamic user profiling for streams of short texts," in *AAAI*, 2018, pp. 5860–5867.

[5] S. Liang, X. Zhang, Z. Ren, and E. Kanoulas, "Dynamic embeddings for user profiling in twitter," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, Jul. 2018. [Online]. Available: https://doi.org/10.1145/3219819.3220043

[6] W. Liang, H. Xie, Y. Rao, R. Y. Lau, and F. L. Wang, "Universal affective model for readers' emotion classification over short texts," *Expert Systems with Applications*, vol. 114, pp. 322–333, Dec. 2018. [Online]. Available: https://doi.org/10.1016/j.eswa.2018.07.027

[7] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013. [Online]. Available: https://doi.org/10.1109/tpami.2013.50

[8] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, Jan. 2015. [Online]. Available: https://doi.org/10.1016/j.neunet.2014.09.003

[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015. [Online]. Available: https://doi.org/10.1038/nature14539

[10] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio, "Generative adversarial nets," in *NeurIPS 2014, December 8-13 2014, Montreal, Quebec, Canada*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds., 2014, pp. 2672–2680.

[11] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2014. [Online]. Available: http://arxiv.org/abs/1312.6114

[12] D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling, "Semi-supervised learning with deep generative models," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'14. Cambridge, MA, USA: MIT Press, 2014, p. 3581–3589.

[13] A. P. Dempster, "Upper and lower probabilities induced by a multivalued mapping," *The Annals of Mathematical Statistics*, vol. 38, no. 2, pp. 325–339, Apr. 1967.

[14] G. Shafer, *A Mathematical Theory of Evidence*. Princeton: Princeton University Press, 1976.

[15] A. P. Dempster, "A generalization of bayesian inference," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 30, no. 2, pp. 205–232, Jul. 1968.

[16] T. Denœux, S. Sriboonchitta, and O. Kanjanatarakul, "Evidential clustering of large dissimilarity data," *Knowledge-Based Systems*, vol. 106, pp. 179–195, Aug. 2016.

[17] G. Shafer, "A mathematical theory of evidence turns 40," *International Journal of Approximate Reasoning*, vol. 79, pp. 7–25, Dec. 2016.

[18] T. Denœux, "40 years of dempster–shafer theory," *International Journal of Approximate Reasoning*, vol. 79, pp. 1–6, Dec. 2016. [Online]. Available: https://doi.org/10.1016/j.ijar.2016.07.010

[19] S. McClean and B. Scotney, "Using evidence theory for the integration of distributed databases," *International Journal of Intelligent Systems*, vol. 12, no. 10, pp. 763–776, Oct. 1997.

[20] V.-N. Huynh, Y. Nakamori, T.-B. Ho, and T. Murai, "Multiple-attribute decision making under uncertainty: the evidential reasoning approach revisited," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 36, no. 4, pp. 804–822, Jul. 2006.

[21] K. Wickramaratna, M. Kubat, and K. Premaratne, "Predicting missing items in shopping carts," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 7, pp. 985–998, Jul. 2009. [Online]. Available: https://doi.org/10.1109/tkde.2008.229

[22] T. L. Wickramarathne, K. Premaratne, M. Kubat, and D. T. Jayaweera, "CoFiDS: A belief-theoretic approach for automated collaborative filtering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 2, pp. 175–189, Feb. 2011. [Online]. Available: https://doi.org/10.1109/tkde.2010.88

[23] Y. Li, J. Chen, and L. Feng, "Dealing with uncertainty: A survey of theories and practices," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 11, pp. 2463–2482, Nov. 2013. [Online]. Available: https://doi.org/10.1109/tkde.2012.179

[24] A.-S. Capelle, O. Colot, and C. Fernandez-Maloigne, "Evidential segmentation scheme of multi-echo MR images for the detection of brain tumors using neighborhood information," *Information Fusion*, vol. 5, no. 3, pp. 203–216, Sep. 2004. [Online]. Available: https://doi.org/10.1016/j.inffus.2003.10.001

[25] Y. Chen, A. B. Cremers, and Z. Cao, "Interactive color image segmentation via iterative evidential labeling," *Information Fusion*, vol. 20, pp. 292–304, Nov. 2014. [Online]. Available: https://doi.org/10.1016/j.inffus.2014.03.007

[26] T. Denœux and L. M. Zouhal, "Handling possibilistic labels in pattern classification using evidential reasoning," *Fuzzy Sets and Systems*, vol. 122, no. 3, pp. 409–424, Sep. 2001.

[27] D. Gruyer, S. Demmel, V. Magnier, and R. Belaroussi, "Multi-hypotheses tracking using the dempster–shafer theory, application to ambiguous road context," *Information Fusion*, vol. 29, pp. 40–56, May 2016. [Online]. Available: https://doi.org/10.1016/j.inffus.2015.10.001

[28] Y. Dan, J. Hongbing, and G. Yongchan, "A robust d–s fusion algorithm for multi-target multi-sensor with higher reliability," *Information Fusion*, vol. 47, pp. 32–44, May 2019. [Online]. Available: https://doi.org/10.1016/j.inffus.2018.06.009

[29] M. C. Florea, A.-L. Jousselme, É. Bossé, and D. Grenier, "Robust combination rules for evidence theory," *Information Fusion*, vol. 10, no. 2, pp. 183–197, Apr. 2009.

[30] E. Lefevre, O. Colot, and P. Vannoorenberghe, "Belief function combination and conflict management," *Information Fusion*, vol. 3, no. 2, pp. 149–162, Jun. 2002.

[31] T. Denoeux, "A k-nearest neighbor classification rule based on dempster-shafer theory," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 25, no. 5, pp. 804–813, May 1995.

[32] Z.-G. Liu, Q. Pan, J. Dezert, and A. Martin, "Combination of classifiers with optimal weight based on evidential reasoning," *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 3, pp. 1217–1230, Jun. 2018. [Online]. Available: https://doi.org/10.1109/tfuzz.2017.2718483

[33] T. Denoeux and M.-H. Masson, "EVCLUS: Evidential clustering of proximity data," *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, vol. 34, no. 1, pp. 95–109, Feb. 2004.

[34] T. Denoeux, "Maximum likelihood estimation from uncertain data in the belief function framework," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 1, pp. 119–130, Jan. 2013. [Online]. Available: https://doi.org/10.1109/tkde.2011.201

[35] D. M. Blei and J. D. Lafferty, "Dynamic topic models," in *Proceedings of the 23rd international conference on Machine learning - ICML '06*. ACM Press, 2006. [Online]. Available: https://doi.org/10.1145/1143844.1143859

[36] T. L. Griffiths and M. Steyvers, "Finding scientific topics," *Proceedings of the National Academy of Sciences*, vol. 101, no. Supplement 1, pp. 5228–5235, Feb. 2004.

[37] K. Ren, J. Qin, Y. Fang, W. Zhang, L. Zheng, W. Bian, G. Zhou, J. Xu, Y. Yu, X. Zhu, and K. Gai, "Lifelong sequential modeling with personalized memorization for user response prediction," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR'19. New York, NY, USA: Association for Computing Machinery, 2019, p. 565–574. [Online]. Available: https://doi.org/10.1145/3331184.3331230

[38] T. Denoeux, "A neural network classifier based on dempster-shafer theory," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 30, no. 2, pp. 131–150, Mar. 2000.

[39] P. Smets, "Data fusion in the transferable belief model," in *Proceedings of the Third International Conference on Information Fusion*, vol. 1. IEEE, 2000, pp. PS21–PS33.

[40] S. Herculano-Houzel, "The human brain in numbers: a linearly scaled-up primate brain," *Frontiers in Human NeuroScience*, vol. 3, 2009. [Online]. Available: https://doi.org/10.3389/neuro.09.031.2009

[41] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, Dec. 1989. [Online]. Available: https://doi.org/10.1162/neco.1989.1.4.541

[42] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

[43] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *Proceedings of the 31st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, E. P. Xing and T. Jebara, Eds., vol. 32, no. 2. Bejing, China: PMLR, 22–24 Jun 2014, pp. 1278–1286. [Online]. Available: https://proceedings.mlr.press/v32/rezende14.html

[44] T. R. Davidson, L. Falorsi, N. D. Cao, T. Kipf, and J. M. Tomczak, "Hyperspherical variational auto-encoders," 2018.

[45] N. Dilokthanakul, P. A. M. Mediano, M. Garnelo, M. C. H. Lee, H. Salimbeni, K. Arulkumaran, and M. Shanahan, "Deep unsupervised clustering with gaussian mixture variational autoencoders," 2017.

[46] E. T. Nalisnick and P. Smyth, "Stick-breaking variational autoencoders," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. [Online]. Available: https://openreview.net/forum?id=S1jmAotxg

[47] J. M. Tomczak and M. Welling, "Vae with a vampprior," 2018.

[48] C. Liu and J. Zhu, "Riemannian stein variational gradient descent for bayesian inference," 2017.

[49] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. B. Tenenbaum, "Deep convolutional inverse graphics network," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, p. 2539–2547.

[50] T. Salimans, D. P. Kingma, and M. Welling, "Markov chain monte carlo and variational inference: Bridging the gap," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML'15. JMLR.org, 2015, p. 1218–1226.

[51] K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, and D. Wierstra, "Draw: A recurrent neural network for image generation," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML'15. JMLR.org, 2015, p. 1462–1471.

[52] K. Sohn, X. Yan, and H. Lee, "Learning structured output representation using deep conditional generative models," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, p. 3483–3491.

[53] J. Walker, C. Doersch, A. Gupta, and M. Hebert, "An uncertain future: Forecasting from static images using variational autoencoders," in *Proceedings of (ECCV) European Conference on Computer Vision*, October 2016, pp. 835 – 851.

[54] S. Odaibo, "Tutorial: Deriving the standard variational autoencoder (vae) loss function," 2019.

[55] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Jun. 2019. [Online]. Available: https://doi.org/10.1109/cvpr.2019.00453

[56] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *CVPR 2020, Seattle, WA, USA, June 13-19, 2020*. IEEE, 2020, pp. 8107–8116. [Online]. Available: https://doi.org/10.1109/CVPR42600.2020.00813

[57] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, "Semantic image synthesis with spatially-adaptive normalization," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Jun. 2019. [Online]. Available: https://doi.org/10.1109/cvpr.2019.00244

[58] S. Menon, A. Damian, S. Hu, N. Ravi, and C. Rudin, "PULSE: self-supervised photo upsampling via latent space exploration of generative models," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*. IEEE, 2020, pp. 2434–2442. [Online]. Available: https://doi.org/10.1109/CVPR42600.2020.00251

[59] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery," in *Lecture Notes in Computer Science*. Springer International Publishing, 2017, pp. 146–157. [Online]. Available: https://doi.org/10.1007/978-3-319-59050-9_12

[60] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. Efros, "Context encoders: Feature learning by inpainting," in *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[61] G. Perarnau, J. van de Weijer, B. Raducanu, and J. M. Álvarez, "Invertible Conditional GANs for image editing," in *NIPS Workshop on Adversarial Training*, 2016.

[62] Y. Shen, J. Gu, X. Tang, and B. Zhou, "Interpreting the latent space of gans for semantic face editing," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle,*

WA, USA, June 13-19, 2020. IEEE, 2020, pp. 9240–9249. [Online]. Available: https://doi.org/10.1109/CVPR42600.2020.00926

[63] C.-H. Lee, Z. Liu, L. Wu, and P. Luo, "MaskGAN: Towards diverse and interactive facial image manipulation," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Jun. 2020. [Online]. Available: https://doi.org/10.1109/cvpr42600.2020.00559

[64] J. Wu, C. Zhang, T. Xue, W. T. Freeman, and J. B. Tenenbaum, "Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, ser. NIPS'16. Red Hook, NY, USA: Curran Associates Inc., 2016, p. 82–90.

[65] Y. Jin, J. Zhang, M. Li, Y. Tian, H. Zhu, and Z. Fang, "Towards the automatic anime characters creation with generative adversarial networks," *CoRR*, vol. abs/1708.05509, 2017. [Online]. Available: http://arxiv.org/abs/1708.05509

[66] Y. Taigman, A. Polyak, and L. Wolf, "Unsupervised cross-domain image generation," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. [Online]. Available: https://openreview.net/forum?id=Sk2Im59ex

[67] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML'16. JMLR.org, 2016, p. 1060–1069.

[68] Y. Wang and H.-Y. Lee, "Learning to encode text as human-readable summaries using generative adversarial networks," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, Oct.-Nov. 2018, pp. 4187–4195. [Online]. Available: https://www.aclweb.org/anthology/D18-1451

[69] T. Qiao, J. Zhang, D. Xu, and D. Tao, "MirrorGAN: Learning text-to-image generation by redescription," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Jun. 2019. [Online]. Available: https://doi.org/10.1109/cvpr.2019.00160

[70] S. Pan, L. Dai, X. Hou, H. Li, and B. Sheng, *ChefGAN: Food Image Generation from Recipes*. New York, NY, USA: Association for Computing Machinery, 2020, p. 4244–4252. [Online]. Available: https://doi.org/10.1145/3394171.3413636

[71] L. Yang, S. Chou, and Y. Yang, "Midinet: A convolutional generative adversarial network for symbolic-domain music generation using 1d and 2d conditions," *CoRR*, vol. abs/1703.10847, 2017. [Online]. Available: http://arxiv.org/abs/1703.10847

[72] K. Balog, "Expertise retrieval," *Foundations and Trends® in Information Retrieval*, vol. 6, no. 2-3, pp. 127–256, 2012.

[73] N. Craswell, A. P. de Vries, and I. Soboroff, "Overview of the trec 2005 enterprise track," in *TREC*, vol. 5, 2005, pp. 199–205.

[74] K. Balog, T. Bogers, L. Azzopardi, M. de Rijke, and A. van den Bosch, "Broad expertise retrieval in sparse data environments," in *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. New York, NY, USA: ACM Press, 2007, pp. 551–558.

[75] D. Estival, T. Gaustad, B. Hutchinson, S. B. Pham, and W. Radford, "Author profiling for english emails," in *In Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*, 2007, pp. 263–272.

[76] W. Deitrick, Z. Miller, B. Valyou, B. Dickinson, T. Munson, and W. Hu, "Gender identification on twitter using the modified balanced winnow," *Communications and Network*, vol. 04, no. 03, pp. 189–195, 2012.

[77] R. Green and J. Sheppard, "Comparing frequency- and style-based features for twitter author identification," in *FLAIRS 2013 - Proceedings of the 26th International Florida Artificial Intelligence Research Society Conference*, 01 2013, pp. 64–69.

[78] S. Bergsma and B. V. Durme, "Using conceptual class attributes to characterize social media users," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Sofia, Bulgaria: Association for Computational Linguistics, Aug. 2013, pp. 710–720.

[79] D. Preoţiuc-Pietro, V. Lampos, and N. Aletras, "An analysis of the user occupational class through twitter content," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*. Beijing, China: Association for Computational Linguistics, Jul. 2015, pp. 1754–1764.

[80] B. Han, P. Cook, and T. Baldwin, "A stacking-based approach to twitter user geolocation prediction," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Sofia, Bulgaria: Association for Computational Linguistics, Aug. 2013, pp. 7–12.

[81] A. Rahimi, T. Cohn, and T. Baldwin, "Twitter user geolocation using a unified text and network prediction model," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Beijing, China: Association for Computational Linguistics, Jul. 2015, pp. 630–636.

[82] X. Zheng, J. Han, and A. Sun, "A survey of location prediction on twitter," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 9, pp. 1652–1671, Sep. 2018. [Online]. Available: https://doi.org/10.1109/tkde.2018.2807840

[83] R. Cohen and D. Ruths, "Classifying political orientation on twitter: It's not easy!" in *Proceedings of the 7th International Conference on Weblogs and Social Media, ICWSM 2013*. The AAAI Press, 01 2013, pp. 91–99.

[84] S. Volkova, G. Coppersmith, and B. V. Durme, "Inferring user political preferences from streaming communications," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Baltimore, Maryland: Association for Computational Linguistics, Jun. 2014, pp. 186–196.

[85] D. Tang, B. Qin, and T. Liu, "Learning semantic representations of users and products for document level sentiment classification," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*. Beijing, China: Association for Computational Linguistics, Jul. 2015, pp. 1014–1023.

[86] D.-V. Vo, J. Karnjana, and V.-N. Huynh, "An integrated framework of learning and evidential reasoning for user profiling using short

texts," *Information Fusion*, vol. 70, pp. 27–42, Jun. 2021. [Online]. Available: https://doi.org/10.1016/j.inffus.2020.12.004

[87] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 05 2003.

[88] T. Hofmann, "Probabilistic latent semantic indexing," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR'99*. ACM Press, 1999, pp. 50–57.

[89] M. Girolami and A. Kabán, "On an equivalence between PLSI and LDA," in *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval - SIGIR'03*. ACM Press, 2003, pp. 433–434.

[90] H. M. Wallach, D. M. Mimno, and A. McCallum, "Rethinking lda: Why priors matter," in *Advances in Neural Information Processing Systems 22*, Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, Eds. Curran Associates, Inc., 01 2009, vol. 23, pp. 1973–1981.

[91] X. Wei, J. Sun, and X. Wang, "Dynamic mixture models for multiple time-series," in *IJCAI 2007, Hyderabad, India, January 6-12, 2007*, M. M. Veloso, Ed., 2007, pp. 2909–2914. [Online]. Available: http://ijcai.org/Proceedings/07/Papers/468.pdf

[92] T. Iwata, S. Watanabe, T. Yamada, and N. Ueda, "Topic tracking model for analyzing consumer purchase behavior," in *Proceedings of the 21st International Jont Conference on Artifical Intelligence*, ser. IJCAI'09. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2009, p. 1427–1432.

[93] G. Heinrich, "Parameter estimation for text analysis," *Technical Report*, pp. 1–31, 01 2005.

[94] K. Murphy, *Machine learning : a probabilistic perspective*. MIT Press, 2013.

[95] Y. Rao, H. Xie, J. Li, F. Jin, F. L. Wang, and Q. Li, "Social emotion classification of short text via topic-level maximum entropy model," *Information & Management*, vol. 53, no. 8, pp. 978–986, Dec. 2016. [Online]. Available: https://doi.org/10.1016/j.im.2016.04.005

[96] J. Pang, Y. Rao, H. Xie, X. Wang, F. L. Wang, T. Wong, and Q. Li, "Fast supervised topic models for short text emotion detection," *IEEE Transactions on Cybernetics*, pp. 1–14, 2019.

[97] J. Feng, Y. Rao, H. Xie, F. L. Wang, and Q. Li, "User group based emotion detection and topic discovery over short text," *World Wide Web*, vol. 23, no. 3, pp. 1553–1587, Dec. 2019. [Online]. Available: https://doi.org/10.1007/s11280-019-00760-3

[98] S. Liang, Z. Ren, Y. Zhao, J. Ma, E. Yilmaz, and M. D. Rijke, "Inferring dynamic user interests in streams of short texts for user clustering," *ACM Transactions on Information Systems*, vol. 36, no. 1, pp. 1–37, Aug. 2017. [Online]. Available: https://doi.org/10.1145/3072606

[99] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543.

[100] P. Cimiano and J. Völker, "Text2onto - a framework for ontology learning and data-driven change discovery," in *Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems (NLDB)*, ser. Lecture Notes in Computer Science, vol. 3513. Alicante, Spain: Springer, Jun. 2005, pp. 227–238.

[101] P. Velardi, S. Faralli, and R. Navigli, "OntoLearn reloaded: A graph-based algorithm for taxonomy induction," *Computational Linguistics*, vol. 39, no. 3, pp. 665–707, 2013.

[102] G. Wohlgenannt, "Leveraging and balancing heterogeneous sources of evidence in ontology learning," in *The Semantic Web. Latest Advances and New Domains*. Springer International Publishing, 2015, pp. 54–68.

[103] W. Wong, W. Liu, and M. Bennamoun, "Ontology learning from text," *ACM Computing Surveys*, vol. 44, no. 4, pp. 1–36, Aug. 2012.

[104] J. Yin and J. Wang, "A dirichlet multinomial mixture model-based approach for short text clustering," *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 08 2014.

[105] S. Rose, D. Engel, N. Cramer, and W. Cowley, "Automatic keyword extraction from individual documents," in *Text Mining*. John Wiley & Sons, Ltd, Mar. 2010, pp. 1–20. [Online]. Available: https://doi.org/10.1002/9780470689646.ch1

[106] R. Mihalcea and P. Tarau, "TextRank: Bringing order into text," in *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Barcelona, Spain: Association for Computational Linguistics, Jul. 2004, pp. 404–411. [Online]. Available: https://www.aclweb.org/anthology/W04-3252

[107] F. Barrios, F. López, L. Argerich, and R. Wachenchauzer, "Variations of the similarity function of textrank for automated summarization," *Proc. Argentine Symposium on Artificial Intelligence, ASAI*, 02 2016.

[108] A. Rajaraman and J. Ullman, "Data mining," in *Mining of Massive Datasets*. Cambridge University Press, 2011, pp. 1–17.

[109] F. Boudin, "pke: an open source python-based keyphrase extraction toolkit," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, Osaka, Japan, December 2016, pp. 69–73. [Online]. Available: http://aclweb.org/anthology/C16-2015

[110] R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, C. Nunes, and A. Jatowt, "YAKE! keyword extraction from single documents using multiple local features," *Information Sciences*, vol. 509, pp. 257–289, Jan. 2020. [Online]. Available: https://doi.org/10.1016/j.ins.2019.09.013

[111] C. Florescu and C. Caragea, "PositionRank: An unsupervised approach to keyphrase extraction from scholarly documents," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2017. [Online]. Available: https://doi.org/10.18653/v1/p17-1102

[112] F. Boudin, "Unsupervised keyphrase extraction with multipartite graphs," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 667–672. [Online]. Available: https://aclanthology.org/N18-2105

[113] W. B. Croft, D. Metzler, and T. Strohman, *Search engines: information retrieval in practice*, 1st ed. Boston: Addison-Wesley, Feb. 2010.

[114] H. Xie, A. Liu, F. L. Wang, T.-L. Wong, X. Liu, and Y. Rao, "Revisit tag-based profiles in the folksonomy: How many tags are sufficient for profiling?" in *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*. IEEE, Feb. 2017, pp. 274–277.

[115] A. Borji, "Pros and cons of GAN evaluation measures," *Computer Vision and Image Understanding*, vol. 179, pp. 41–65, Feb. 2019. [Online]. Available: https://doi.org/10.1016/j.cviu.2018.10.009

[116] B. Dai and D. P. Wipf, "Diagnosing and enhancing VAE models," in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. [Online]. Available: https://openreview.net/forum?id=B1e0X3C9tQ

[117] L. Girin, S. Leglaive, X. Bie, J. Diard, T. Hueber, and X. Alameda-Pineda, "Dynamical variational autoencoders: A comprehensive review," *CoRR*, vol. abs/2008.12595, 2020. [Online]. Available: https://arxiv.org/abs/2008.12595

[118] H. Shao, Z. Xiao, S. Yao, A. Zhang, S. Liu, and T. Abdelzaher, "Controlvae: Tuning, analytical properties, and performance analysis," 2020.

[119] L. Theis, A. van den Oord, and M. Bethge, "A note on the evaluation of generative models," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2016. [Online]. Available: http://arxiv.org/abs/1511.01844

[120] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, Apr. 2004. [Online]. Available: https://doi.org/10.1109/tip.2003.819861

[121] J. Snell, K. Ridgeway, R. Liao, B. D. Roads, M. C. Mozer, and R. S. Zemel, "Learning to generate images with perceptual similarity metrics," 2017.

[122] F. Juefei-Xu, V. N. Boddeti, and M. Savvides, "Gang of gans: Generative adversarial networks with maximum margin ranking," 2017.

[123] S. Xiang and H. Li, "On the effects of batch and weight normalization in generative adversarial networks," 2017.

[124] X. Huang, Y. Li, O. Poursaeed, J. Hopcroft, and S. Belongie, "Stacked generative adversarial networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Jul. 2017. [Online]. Available: https://doi.org/10.1109/cvpr.2017.202

[125] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. Metaxas, "StackGAN: Text to photo-realistic image synthesis with stacked generative adversarial networks," in *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2017. [Online]. Available: https://doi.org/10.1109/iccv.2017.629

[126] C. Xiao, B. Li, J.-Y. Zhu, W. He, M. Liu, and D. Song, "Generating adversarial examples with adversarial networks," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, ser. IJCAI'18. AAAI Press, 2018, p. 3905–3911.

[127] A. Srivastava, L. Valkov, C. Russell, M. U. Gutmann, and C. Sutton, "Veegan: Reducing mode collapse in gans using implicit variational learning," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 3310–3320.

[128] Z. Lin, A. Khetan, G. Fanti, and S. Oh, "PacGAN: The power of two samples in generative adversarial networks," *IEEE Journal on Selected Areas in Information Theory*, vol. 1, no. 1, pp. 324–335, May 2020. [Online]. Available: https://doi.org/10.1109/jsait.2020.2983071

[129] R. Zafarani and H. Liu, "Connecting users across social media sites," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, Aug. 2013. [Online]. Available: https://doi.org/10.1145/2487575.2487648

[130] ——, "Users joining multiple sites: Distributions and patterns," in *Proceedings of the Eighth International Conference on Weblogs and Social Media, ICWSM 2014, Ann Arbor, Michigan, USA, June 1-4, 2014*, E. Adar, P. Resnick, M. D. Choudhury, B. Hogan, and A. H. Oh, Eds. The AAAI Press, 2014.

[131] R. Zafarani, L. Tang, and H. Liu, "User identification across social media," *ACM Transactions on Knowledge Discovery from*

Data, vol. 10, no. 2, pp. 1–30, Oct. 2015. [Online]. Available: https://doi.org/10.1145/2747880

[132] L. Tang, H. Liu, and J. Zhang, "Identifying evolving groups in dynamic multimode networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 1, pp. 72–85, Jan. 2012. [Online]. Available: https://doi.org/10.1109/tkde.2011.159

[133] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," 2015.

[134] S. Herdade, A. Kappeler, K. Boakye, and J. Soares, "Image captioning: Transforming objects into words," 2020.

Appendices

This appendix provides the derivation mass function combination in the case that there are more than 2 pieces of evidence as mentioned in Chapter 3.

Problem. Let $\mathcal{L} = \{m_1, m_2, ..., m_L\}$ be a set of mass functions over the same frame of discernment Ω, where Ω is a set of all possible answer to a question ($\Omega = \{ak_1, ak_2, ..., ak_V\}$). Each m_i is a mass function corresponding to one piece of evidence. Prove that the overall mass function when fusing all m_i is determined by:

$$m^{(1,2,..,M)}(\{ak_t\}) = \frac{1}{\mathcal{K}}\left[\prod_{i=1}^{M} m_i(\{ak_t\}) + \sum_{k=1}^{M-1}\sum_{\substack{S_j\subseteq\{1,...,M\}\\|S_j|=k}}\prod_{u\in S_j} m_u(\{ak_t\})\prod_{\substack{v=1\\v\notin S_j}}^{V} m_v(\Omega)\right], 1 \le t \le V \tag{5.1}$$

$$m^{(1,2,...,M)}(\Omega) = \frac{1}{\mathcal{K}}\sum_{i=1}^{M} m_i(\Omega) \tag{5.2}$$

where $m^{(1,2,...,M)}\big(\{ak_t\}\big)$ is a mass value

$$\mathcal{K} = \sum_{t=1}^{V}\left[\prod_{i=1}^{M} m_i(\{ak_t\}) + \sum_{k=1}^{M-1}\sum_{\substack{S_j\subseteq\{1,...,M\}\\|S_j|=k}}\prod_{u\in S_j} m_u(\{ak_t\})\prod_{\substack{v=1\\v\notin S_j}}^{V} m_v(\Omega) + \prod_{i=1}^{M} m_i(\Omega)\right] \tag{5.3}$$

assigned for a particular subset $\{ak_t\} \subseteq \Omega$, $\forall t \in \big[1, V\big]$, and $\mathcal{K}$ is the normalization factor when combining masses from M different sources.

Proof.

Because $\mathcal{K}$ is just a normalization factor, we ignore it during proving process. According to the fomula for $m(\Omega)$, observing that the only way to get mass value for the set Ω is to multiply the one in each source, so that:

$$m^{(1,2,...,M)}(\Omega) = \prod_{i=1}^{M} m_i(\Omega) \tag{5.4}$$

We now give a proof for $m^{(1,2,...,M)}\big(\{ak_q\}\big)$ by induction as below:

Basecase: $M = 1$, we get:

$$m\big(\{ak_t\}\big) = m_1\big(\{ak_t\}\big),\ 1 \le t \le V \tag{5.5}$$

$$m(\Omega) = m_1(\Omega) \tag{5.6}$$

$$\text{These formula are valid} \tag{5.7}$$

Hypothesis: assume that is valid for $M = n$, we have:

$$m^{(1,2,..,n)}(\{ak_t\}) = \frac{1}{\mathcal{K}}\left[\prod_{i=1}^{n} m_i(\{ak_t\}) + \sum_{k=1}^{n-1}\sum_{\substack{S_j\subseteq\{1,\dots,M\}\\|S_j|=k}}\prod_{u\in S_j} m_u\big(\{ak_t\}\big)\prod_{\substack{v=1\\v\notin S_j}}^{V} m_v(\Omega)\right],\ 1 \le t \le V \tag{5.8}$$

$$m^{(1,2,\dots,n)}(\Omega) = \frac{1}{\mathcal{K}}\sum_{i=1}^{n} m_i(\Omega) \tag{5.9}$$

$$\mathcal{K} = \sum_{t=1}^{V}\left[\prod_{i=1}^{n} m_i(\{ak_t\}) + \mathcal{C} + \prod_{i=1}^{n} m_i(\Omega)\right] \tag{5.10}$$

Inductive Step: combine n sources with the $(n+1)^{th}$ source gives us:

$$
\begin{aligned}
m^{(1,2,..,n+1)}(\{ak_t\}) &= \left[\prod_{i=1}^{n} m_i(\{ak_t\}) + \sum_{k=1}^{n-1} \sum_{\substack{S_j\subseteq\{1,\dots,M\}\\|S_j|=k}} \prod_{u\in S_j} m_u\big(\{ak_t\}\big) \prod_{\substack{v=1\\v\notin S_j}}^{V} m_v(\Omega)\right] * m_{n+1}(\{ak_t\}) \\
&+ \left[\prod_{i=1}^{n} m_i(\{ak_t\}) + \sum_{k=1}^{n-1} \sum_{\substack{S_j\subseteq\{1,\dots,M\}\\|S_j|=k}} \prod_{u\in S_j} m_u\big(\{ak_t\}\big) \prod_{\substack{v=1\\v\notin S_j}}^{V} m_v(\Omega)\right] * m_{n+1}(\Omega) \\
&+ \prod_{i=1}^{n} m_i(\Omega) * m_{n+1}(\{ak_t\}) \\
&= \prod_{i=1}^{n+1} m_i(\{ak_t\}) \\
&+ m_{n+1}(\{ak_t\}) \sum_{k=1}^{n-1} \sum_{\substack{S_j\subseteq\{1,\dots,M\}\\|S_j|=k}} \prod_{u\in S_j} m_u\big(\{ak_t\}\big) \prod_{\substack{v=1\\v\notin S_j}}^{V} m_v(\Omega) \\
&+ m_{n+1}(\Omega) \prod_{i=1}^{n} m_i\big(\{ak_t\}\big) \\
&+ m_{n+1}(\Omega) \sum_{k=1}^{n-1} \sum_{\substack{S_j\subseteq\{1,\dots,M\}\\|S_j|=k}} \prod_{u\in S_j} m_u\big(\{ak_t\}\big) \prod_{\substack{v=1\\v\notin S_j}}^{V} m_v(\Omega) \qquad (5.11) \\
&+ m_{n+1}\big(\{ak_t\}\big) \prod_{i=1}^{n} m_i(\Omega) \qquad (5.12)
\end{aligned}
$$

Now expanding the term in (5.11), then taking the sum with (5.12) yields,

$$
\begin{aligned}
(5.12) + (5.11) &= m_{n+1}\big(\{ak_t\}\big) \prod_{i=1}^{n} m_i(\Omega) + m_{n+1}(\Omega) \sum_{\substack{S_j\subseteq\{1,\dots,n\}\\|S_j|=1}} \prod_{u\in S_j} m_u\big(\{ak_t\}\big) \prod_{\substack{v=1\\v\notin S_j}}^{V} m_v(\Omega) \\
&+ m_{n+1}(\Omega) \sum_{k=2}^{n-1} \sum_{\substack{S_j\subseteq\{1,\dots,M\}\\|S_j|=k}} \prod_{u\in S_j} m_u\big(\{ak_t\}\big) \prod_{\substack{v=1\\v\notin S_j}}^{V} m_v(\Omega) \\
&= \sum_{\substack{S_j\subseteq\{1,\dots,n+1\}\\|S_j|=1}} \prod_{u\in S_j} m_u\big(\{ak_t\}\big) \prod_{\substack{v=1\\v\notin S_j}}^{V} m_v(\Omega) + \qquad (5.13) \\
&= m_{n+1}(\Omega) \sum_{k=2}^{n-1} \sum_{\substack{S_j\subseteq\{1,\dots,n\}\\|S_j|=k}} \prod_{u\in S_j} m_u\big(\{ak_t\}\big) \prod_{\substack{v=1\\v\notin S_j}}^{V} m_v(\Omega) \qquad (5.14)
\end{aligned}
$$

Similarly, continuing the same process with other terms ends up with:

$$
\begin{aligned}
m^{(1,2,..,n+1)}(\{ak_t\}) &= \prod_{i=1}^{n} m_i\left(\{ak_t\}\right) \\
&+ \sum_{\substack{S_j \subseteq \{1,\ldots,n+1\} \\ |S_j|=1}} \prod_{u \in S_j} m_u\big(\{ak_t\}\big) \prod_{\substack{v \in 1 \\ v \notin S_j}}^{V} m_v(\Omega) \\
&+ \sum_{\substack{S_j \subseteq \{1,\ldots,n+1\} \\ |S_j|=2}} \prod_{u \in S_j} m_u\big(\{ak_t\}\big) \prod_{\substack{v \in 1 \\ v \notin S_j}}^{V} m_v(\Omega) \\
&\ldots \\
&+ \sum_{\substack{S_j \subseteq \{1,\ldots,n+1\} \\ |S_j|=n}} \prod_{u \in S_j} m_u\big(\{ak_t\}\big) \prod_{\substack{v \in 1 \\ v \notin S_j}}^{V} m_v(\Omega) \\
&= \prod_{i=1}^{n+1} m_i\left(\{ak_t\}\right) + \sum_{k=1}^{n} \sum_{\substack{S_j \subseteq \{1,\ldots,n+1\} \\ |S_j|=k}} \prod_{u \in S_j} m_u\big(\{ak_t\}\big) \prod_{\substack{v=1 \\ v \notin S_j}}^{V} m_v(\Omega), \ (1 \leq t \leq V)
\end{aligned}
\tag{5.15}
$$

□