Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)

On Conceptual Labeling of a Bag of Words

Xiangyan Sun[§], Yanghua Xiao[§*], Haixun Wang[†], Wei Wang[§]
[§]School of Computer Science, Shanghai Key Laboratory of Data Science
Fudan University, Shanghai, China
[†]Google Research, USA
{xiangyansun, shawyh, weiwang1}@fudan.edu.cn, haixun@google.com

Abstract

In natural language processing and information retrieval, the bag of words representation is used to implicitly represent the meaning of the text. Implicit semantics, however, are insufficient in supporting text or natural language based interfaces, which are adopted by an increasing number of applications. Indeed, in applications ranging from automatic ontology construction to question answering, explicit representation of semantics is starting to play a more prominent role. In this paper, we introduce the task of conceptual labeling (CL), which aims at generating a minimum set of conceptual labels that best summarize a bag of words. We draw the labels from a data driven semantic network that contains millions of highly connected concepts. The semantic network provides meaning to the concepts, and in turn, it provides meaning to the bag of words through the conceptual labels we generate. To achieve our goal, we use an information theoretic approach to trade-off the semantic coverage of a bag of words against the minimality of the output labels. Specifically, we use Minimum Description Length (MDL) as the criteria in selecting the best concepts. Our extensive experimental results demonstrate the effectiveness of our approach in representing the explicit semantics of a bag of words.

1 Introduction

Many natural language processing and information retrieval tasks adopt the bag of words model as a representation of text. For an increasing number of advanced applications, this simplified representation is insufficient because words and phrases are regarded as atomic symbols. In this paper, we focus on the problem of *conceptual labeling*. Specifically, given a set of words or phrases, our goal is to generate a small set of labels that best summarize the set of words or phrases. Here are a few examples:

1. `china`, `japan`, `india`, `korea` → *asian country*
2. `dinner`, `lunch`, `food`, `child`, `girl` → *meal*, *child*
3. `bride`, `groom`, `dress`, `celebration` → *wedding*

For human beings, the labels on the right hand side are the concepts that come to our mind when we see the words and phrases on the left hand side. We know that, in the first example, *asian country* is a better label than *country*, because *asian country* is more specific while having the same coverage as *country* for the input words. In the second example, we know that the two concepts *meal* and *child* better summarize the input than a single concept *object*. The last example is different from the first two because there is no isA relation between any input word and *wedding*, yet *wedding* is the concept that comes to our mind when we see those words.

The goal of this paper is to generate high quality labels to summarize a bag of words. The quality of the labels are measured by their minimality and coverage. Furthermore, the labels themselves are not atomic symbols. Rather, they are nodes in a fine-grained, highly connected, usage-based semantic network. As a result, machines, not just humans, will be able to understand the meaning of the labels. A bag of words, after being summarized, become objects that machines understand and are able to operate on.

1.1 Applications

Conceptual labeling enables machines to comprehend a set of words. We envision many applications for conceptual labeling, including the following:

- **Topic modeling.** Topic modeling [Blei, 2012] is widely used for discovering latent topics in a collection of documents. However, a topic is a bag of words that do not have explicit semantics. Not only is it hard for humans to interpret the topics, machines also have a very limited capability in using them to interact with other data, as a topic is nothing more than a distribution of meaningless symbols. Conceptual labeling turns each topic into a small set of meaningful concepts grounded in a large semantic network, thus enabling intelligent use of the topics in downstream applications.

[*]Correspondence author. This paper was supported by the National NSFC(No.61472085, 61171132, 61033010), by National Key Basic Research Program of China under No.2015CB358800, by Shanghai STCF under No.13511505302, by NSF of Jiangsu Prov. under No. BK2010280, and by Shanghai Science and Technology Development Funds(13dz2260200, 13511504300).

- **Language understanding.** Much manual work has been devoted to labeling natural language sentences to understand the usage of language. FrameNet [Baker *et al.*, 1998], for example, describes semantic frames, which are mainly about the roles of verbs and their arguments. But even with the labeled sentences, semantic role labeling [Palmer *et al.*, 2010] is hard because there is no mechanism to generalize from a set of words to concepts, and then from concepts to other related words. In other words, it is difficult to assign roles to words that have never appeared in the labeled data. With conceptual labeling, we may for example, summarize verb *eat*'s direct objects `apple`, `breakfast`, `pork`, `beef`, `bullet`[1], ... into a small set of concepts, such as *fruit*, *meal*, *meat*, *bullet*, which in turn connects to all things "edible" in the underlying semantic network.

1.2 Concepts

We convert a set of words and phrases to a small number of labels, each of which is a **concept** that machines understand. A very important question is, what is the difference between a word and a concept, and why are concepts important?

We use an example to illustrate the difference. In the bag-of-words representation, `fruit` is an atomic symbol, and it is independent from other symbols. If we represent a bag of words by a vector, then `fruit` corresponds to a column, which is independent from other columns, including `vegetable` and `meat`. The concept *fruit*, however, is different. It has subconcepts such as *tropical fruit*, instances such as `apple`, superconcepts such as *food*, attributes such as *acidity*. Furthermore, it is often a direct object of verbs such as *eat*, it is often modified by concepts such as *fresh*, and it often modifies concepts such as *health benefit*, and so on. In the mind of a human being, a concept triggers a network of other concepts, and such a network forms the foundation of cognition.

But in order for machines to understand concepts, they must also have access to a concept network like the one in a human mind. How is this possible? In recent years, much effort has been devoted to building knowledge bases and semantic networks. Some of them, such as WordNet [Miller, 1995], Cyc [Lenat, 1995], DBpedia [Auer *et al.*, 2007], and Freebase [Bollacker *et al.*, 2008], are created by human experts or community efforts. Others, such as KnowItAll [Etzioni *et al.*, 2004], NELL [Carlson *et al.*, 2010], and Probase [Wu *et al.*, 2012], are created by data driven approaches. Because information in data driven knowledge bases and semantic networks is usage based, it is particularly useful for natural language understanding. More specifically, data driven semantic networks are special in the following aspects:

1. It has a large, fine-grained concept space. For example, Probase contains millions of concepts. This makes it possible to approximate the concepts in a human mind. To see why we need fine-grained concepts, consider two inputs {`china`, `india`, `germany`, `usa`} and {`china`, `india`, `japan`, `singapore`}. The best concept for the former is *country*, and the best concept for the latter is *asian country*. Without fine granularity concepts such as *asian country*, machines will not be able to summarize the second case as well as humans do.
2. It contains knowledge that is not black or white, but usage based. Knowledge therein is associated with various weights and probabilities, which enable inferencing. For instance, although `robin` and `penguin` are both *birds*, they are not the same, in the sense that a `robin` is a more typical *bird* than a *penguin*, and this intuition is captured by the semantic network in the following form: $P(\texttt{robin}|\textit{bird}) > P(\texttt{penguin}|\textit{bird})$. Clearly, such probabilities are important to a wide range of inferencing tasks.

In this paper, we use Probase[2] to provide us fine-grained concepts and their statistics. Probase is acquired from 1.68 billion web pages. It extracts *isA* relations from sentences matching Hearst patterns [Hearst, 1992]. For example, from the sentence `... presidents such as Obama ...`, it extracts evidence for the claim that `Obama` is an instance of the concept *president*. The core version of Probase contains 3,024,814 unique concepts, 6,768,623 unique instances, and 29,625,920 isA relations.

1.3 Challenge

Our main challenge is to measure the "goodness" of the labels we assign to a bag of words. Generally, a good set of labels should satisfy the following criteria. The two criteria are actually conflicting to each other. How to find the best trade-off between them lies at the core of conceptual labeling.

- *Coverage*. The conceptual labels should cover as many words and phrases in the input as possible, otherwise information in the input is lost. For example, assume a photo is labeled by a bag of words {`vehicle`, `car`, `bicycle`, `road`, `pedestrian`}. If we simply summarize the input as *vehicle*, then we incur information loss as the picture was probably taken on a road with pedestrians nearby.
- *Minimality*. Psychology research shows that an input triggers concepts in a human mind in the most *economical* way. Conceptual labeling aims at the same effect. That is, we want to find the minimal number of conceptual labels to label the words. For example, for {`putty`, `kitten`, `small dog`}, a single conceptual label *pet* is sufficient to characterize the input. Although labels such as {*dog*, *cat*} can also cover the meaning of the input, they are less efficient or economical.

1.4 Paper Organization

The rest of this paper is organized as follows. Section 2 gives an overview of our approach. Section 3 presents in detail our framework for conceptual labeling. We present experimental analysis in Section 4, and conclude in Section 5.

2 Overview

In this section, we describe the minimum description length (MDL) principle and give a brief overview of our approach.

[1] as in "eat a bullet"

[2] Probase data is available at http://probase.msra.cn/dataset.aspx

Minimum Description Length. MDL [Rissanen, 1978] provides an efficient scheme for encoding and compressing data. In general, the more *regularities* in the data, the less bits we need to code it. The ultimate goal of data compression is to find such regularities. Thus, data compression has a natural connection to model selection: Better models tend to better fit the inherent structure of the data, and as a result, the data could be compressed more.

More formally, let X denote the data to be encoded and let $L(x)$ denote the code length of data item $x \in X$. It is well known that for every probabilistic distribution $P(x)$ where $x \in X$, there exists a code for X such that for every $x \in X$, $L(x) = -\log P(x)$. To encode an arbitrary data item x, we usually have two options:

1. Encode the data directly. The code length is $L(x) = -\log P(x)$ bits.

2. Assume a model M captures the regularity in the data. We encode the data with the help of M in $L(M) + L(x|M)$ bits, where $L(M)$ is the number of bits for encoding the model, and $L(x|M) = -\log P(x|M)$ for encoding the data given the model.

If the model M fits the data nicely, the second option may result in a code length that is significantly shorter when we encode a large set of data items $X = \{x_1, x_2, ..., x_n\}$.

Overview of Our Approach. Given a bag of words or phrases X, an external semantic network $G(V_G, E_G)$, where each node $v \in V_G$ is a concept, and each edge $e \in E_G$ is a relationship between two concepts, we want to find the *best* concepts $C \subseteq X \cup V_G$ that summarize X. The key problem is to measure the *quality* of the concepts, and as we mentioned, the measure should consider both *coverage* and *minimality*.

In our work, we use MDL to measure the quality of the concepts. Our goal is to minimize the overall code length for encoding the bag of words. The rationale of using MDL is the following. First, *MDL achieves the best trade-off between coverage and minimality*. We encode all the words so the coverage is good. A concept that covers more words is more likely to be selected since it may significantly decrease the code length. With more concepts, the cost of encoding individual instances decreases, but the cost of encoding more concepts increases. Second, *MDL avoids the problem of deciding the number of concepts in advance*. Traditional approaches (topic modeling and conceptualization) may produce a set of relevant labels but do not provide a measure of how many of them are sufficient. Third, *the results produced by MDL are interpretable*. Because each word is encoded independently, and we know which words belong to which concept. Fourth, as we will show, *the trade-off between coverage and minimality is adjustable*.

3 Conceptual Labeling

In this section, we introduce our conceptual labeling mechanism. We first introduce a basic method that infers a single concept from a bag of words. Then we extend the method to infer multiple concepts.

3.1 Using Knowledge

The labels we assign to a bag of words are grounded in Probase, a data driven semantic network. Probase contains many relationship among concepts, and we mainly use two relationships, namely, the isA relationship and the isPropertyOf (attribute) relationship.

The IsA Relationship. It is obvious that we need isA relationships in summarization. For instance, we may summarize words such as `kitten` and `puppy` to the concept of *pet*. Together with the isA relationship, the semantic network also provides the *typicality* score, which plays an important role in enabling us to select the right concepts. Typicality is defined as follows:

$$P(e|c) = \frac{n(c,e)}{\sum_{e_i} n(c,e_i)} \qquad P(c|e) = \frac{n(c,e)}{\sum_{c_i} n(c_i,e)} \tag{1}$$

where $n(c, e)$ is the frequency of c and e occurring in a syntactic pattern for isA relationship. Intuitively, typicality measures how likely we think of an instance (or a concept) when we are given a concept (or an instance). For example, given concept *pet*, people are more likely to think of a `kitten` than a `monkey`, and this is embodied by $P(\texttt{kitten}|\textit{pet}) > P(\texttt{monkey}|\textit{pet})$ in Probase. Besides typicality, we also need the prior probability of a concept or an instance. We approximate them as follows:

$$P(c) = \frac{\sum_e n(c,e)}{\sum_{(c,e)} n(c,e)} \qquad P(e) = \frac{\sum_c n(c,e)}{\sum_{(c,e)} n(c,e)} \tag{2}$$

The isPropertyOf (Attribute) Relationship. We also need attribute relationships in summarizing a bag of words. For example, {`population, president, location`} triggers the concept `country`, although none of the words has isA relationship with `country`. Probase provides typicality scores $P(c|a)$ and $P(a|c)$, which indicate how likely an attribute a triggers a concept c and vice versa:

$$P(a|c) = \frac{n'(c,a)}{\sum_{a_i} n'(c,a_i)} \qquad P(c|a) = \frac{n'(c,a)}{\sum_{c_i} n'(c_i,a)} \tag{3}$$

where $n'(c, a)$ is the frequency of c and a occur in some syntactic pattern that denotes the isPropertyOf relationship. We will incorporate attribute typicality in our MDL framework for inferencing concepts from a bag of words.

3.2 Labeling a BoW by a Single Concept

Assume a bag of words X invokes a single concept. The code length to encode X with concept c is the following:

$$CL(X,c) = L(c) + L(X|c) = L(c) + \sum_{x_i \in X} L(x_i|c) \tag{4}$$

where $L(c)$ is the code length to describe c alone, and $L(x_i|c)$ is the code length to describe x_i with the prior knowledge of c. Based on the MDL principle, we have

$$CL(X,c) = -\log P(c) + \sum_{x_i \in X} -\log P(x_i|c) \tag{5}$$

where $P(c)$ is the prior probability defined in Eq 2 and $P(x_i|c)$ is the typicality defined in Eq 1 (For now we assume x_i is an instance of c, which will be generalized later.), and our objective is to find the concept that minimizes the description length, i.e.,

$$c^* = \arg\min_c CL(X, c) \tag{6}$$

Next, we show that in this case, the MDL model is equivalent to a Bayesian model [Song *et al.*, 2011]. We have:

$$\begin{aligned} c^* &= \arg\min_c (-\log P(c) + \sum_{x_i \in X} -logP(x_i|c)) \\ &= \arg\min_c -\log(P(c) \prod_{x_i \in X} P(x_i|c)) \\ &= \arg\max_c P(c) \prod_{x_i \in X} P(x_i|c) \\ &= \arg\max_c P(c)P(X|c) = \arg\max_c P(c|X) \end{aligned} \tag{7}$$

The last row is obtained under the independence assumption.

3.3 Labeling a BoW by Multiple Concepts

In many cases, fitting all data items using a single concept might not lead to an optimal solution. For example, {`dinner`, `lunch`, `food`, `child`, `girl`} contains two obvious concepts (*meal* and *child*), and forcing to use a single concept (say *object*) to summarize the input is not going to be optimal.

To address this problem, we extend the coding scheme to multiple models (concepts). Let $\mathcal{C}$ be a model class that contains a finite set of models. We may encode each data item x using the model that gives the maximal posterior likelihood, i.e. $\arg\max_{c \in \mathcal{C}} P(x|c)$. This leads to the shortest code length for x. However, to *decode* the data, we also need to know which model is used to encode x. We describe two possible schemes, namely *two-part code* and *universal code*, for this purpose.

Two-part code. For a word encoded by concept c_i, we also encode the index i. Since we have overall $|C|$ concepts, each index can be encoded with $\log |C|$ bits. Applying the principles of MDL, we have:

$$CL(X, C) = L(C) + L(X|C) = \sum_{c_i \in C} L(c_i) + \sum_{x_i \in X} L^*(x_i|C) \tag{8}$$

where $L^*(x|C)$ is the code length for encoding individual word x given the prior knowledge of C:

$$L^*(x|C) = \log |C| + \min_{c \in C} L(x|c) \tag{9}$$

The input may contain *outliers* that should not be summarized to concepts. For example, {`apple`, `banana`, `breakfast`, `dinner`, `pork`, `beef`, `bullet`} are direct objects of the verb *eat*. We may summarize them into concept {*fruit*, *meal*, *meat*} except for the last word `bullet`. We need to make a choice: *either encode the outlier independently* or *encode it with some concept c*. We use the MDL principle to make the choice. That is, we calculate the code lengths yielded by the two options and select the one with the shorter length. This leads to a new definition of $L^*(x|C)$:

$$L^*(x|C) = \min \begin{cases} L(x), & \text{encode directly} \\ \log |C| + L(x|c), & \text{encode using } c \in C \end{cases} \tag{10}$$

Because each word is encoded independently, the combination of local optimums guarantees the global optimum. Using this scheme each word (x) will be assigned to the concept c which has the maximal posterior probability $P(x|c)$.

Universal code. Alternatively, we may generate a *universal model*, which mixes all the models into one model. For example, we can create a universal model based on occurrence probability, i.e. $P(x|\mathcal{C}) = \sum_{c \in \mathcal{C}} P(x|c)P(c)$. The *regret* measure [Shtar'kov, 1987] is used to evaluate different universal models. For a given data item x, the regret for a universal model $P(x|\mathcal{C})$ relative to the original model class $\mathcal{C}$, is defined as:

$$R(x, P) = -\log P(x|\mathcal{C}) - \min_{c \in \mathcal{C}} \{-\log P(x|c)\} \tag{11}$$

Intuitively, $R(x, P)$ is the additional number of bits to encode x using distribution P compared to using optimal maximum likelihood model. The best universal model should minimize the maximal additional bits over the entire data space, that is

$$\min_P \max_{x \in \mathcal{X}} R(x, P) \tag{12}$$

where $\mathcal{X}$ is the data space that individual data x resides in.

It was shown [Shtar'kov, 1987] that the *normalized maximum likelihood* model achieves the minimum. The normalized maximum likelihood is defined as:

$$P_{NML}(x|\mathcal{C}) = \frac{\hat{P}(x|\mathcal{C})}{\sum_{x \in \mathcal{X}} \hat{P}(x|\mathcal{C})} \tag{13}$$

where $\hat{P}(x|\mathcal{C})$ is the maximal posterior likelihood of x using a model from $\mathcal{C}$, i.e. $\hat{P}(x|\mathcal{C}) = \max_{c \in \mathcal{C}} P(x|c)$.

Using the normalized maximum likelihood code, we can reformulate $L^*(x|C)$ in Eq 10 as:

$$\begin{aligned} L^*(x|C) &= \min(L(x), -\log P_{NML}(x|C)) \\ &= \min(L(x), -\log \frac{\hat{P}(x|C)}{\sum_{x'} \hat{P}(x'|C)}) \\ \hat{P}(x|C) &= \max_{c \in C} P(x|c) \end{aligned} \tag{14}$$

Similar to two-part code, each word x will be assigned to the concept c that has the maximal posterior probability $P(x|c)$.

3.4 Integrating Attributes

In our MDL model, we use $P(x|c)$ to characterize the relationship between a concept c and an input word x. Up to now, we have assumed that the relationship is the isA relationship, and $P(x|c)$ is defined as in Eq 1. However, the input may contain words that are attributes or properties of a concept, as in {`population, president, location`}, which triggers the concept *country*.

To incorporate attributes, we combine the isA and the isPropertyOf relations to a unified probabilistic model. In practice, it is rare that an input word is both an instance and an attribute of a concept. As in [Song *et al.*, 2011], we combine the typicality using a noisy-or model:

$$P(c|x) = 1 - (1 - P_e(c|x))(1 - P_a(c|x)) \tag{15}$$

where $P_e(c|x)$ denotes the isA typicality as defined in Eq 1, and $P_a(c|x)$ denotes the attribute typicality as defined in Eq 3. Intuitively, $P(c|x)$ is the likelihood that the word x invokes concept c, by being either its instance or attribute. The reversed typicality $P(x|c)$ is inferred using the Bayes rule: $P(x|c) = P(c|x)P(x)/P(c)$.

3.5 Tradeoff between Coverage and Minimality

In practice, it may be more desirable to limit the number of concepts, or to generate more concepts for better coverage of meaning. We thus extend our model to add an adjustable parameter for balancing the importance of concepts and tags. We reformulate the final MDL measure to:

$$\begin{aligned} C^* &= \arg\min_{C} \alpha\, L(C) + (1-\alpha) L(X|C) \\ &= \arg\min_{C} \alpha \sum_{c_i \in C} L(c_i) + (1-\alpha) \sum_{x_i \in X} L^*(x_i|C) \end{aligned} \tag{16}$$

where X is the input, C is set of concepts used to encode the input, $L^*(x|C)$ is the code length of individual word, depending on whether two-part code or NML code is used, α is a parameter that can be used to tradeoff coverage and minimality. By default $\alpha = 0.5$. A larger α value indicates the description length of concepts are weighted higher than input words, thus fewer concepts will be generated, vice versa.

3.6 Search Strategy

We have shown how to measure the goodness to use a given set of concepts for the labeling. However, we still need a method to find the best concept set. Since Probase has millions of concepts and relations, exhaustive enumeration over this space is costly. On the other hand, the MDL based measure does not have any desired properties (such as, sub-modularity) that allows an efficient pruning of the search space. Hence, we resort to a greedy heuristic for the search.

We first find all hypernyms of input words. We discard concepts which only contain a very small number of instances. They are unlikely to be good conceptual labels since they only cover a few instances in Probase. Examples include *small gathering*, *big picture* and *emotional activity*. The hypernyms after the filtering constitute the candidate concept set (denote as S). Let C be the current selected concept set. We first set C as empty. Then, we start an iterative procedure. Within each iteration, we try the following three types of operations.

- Move an unused concept from $S \setminus C$ to C
- Remove a concept in C
- Replace a concept in C with another one in $S \setminus C$

For each type of operation, we exhaustively try each possible realization. For example, for the removal of a concept from C, we try the removal of each concept in C. We use the operation yielding the maximal decrease of the description length to materialize the update on the current concept set C. The iteration terminates when the description length can not be decreased any more. Since the code length is monotonically decreasing through each iteration, this algorithm is guaranteed to converge.

For each iteration, the most costly operation is replacement, which has overall $|C||S|$ realizations. In reach realization, we need to compute the MDL measure, which costs $O(|C| + |C||X|)$ time for two-part code, where X is the input word set. Let t be the number of iterations needed for convergence. The overall time complexity is $O(t|C|^2|S|(1 + |X|))$. Since $|C|$ increases at most by one after each iteration, t is the upper bound of $|C|$. Thus, we have the complexity as $O(t^3|S|(1 + |X|))$ for two-part code. For NML code, it is the same except for computing the MDL measure. As in Eq 14, we need to enumerate all entities of concepts in S to compute the denominator. The time complexity for computing MDL measure is $O(|C|D_S + |C||X|)$, where D_S is the sum of degrees of concepts in S. Hence, the overall complexity for NML code is $O(t^3|S|(D_S + |X|))$. In our experiments, $|X|$ is less than 30, and $|S|$ is several hundreds. Our experimental results show that the algorithm usually converges within 10 iterations, implying $t \leq 10$. As Probase is a relatively sparse knowledge base (the average degree is less than 10), D_S is also small (about several thousands). Hence, our algorithm is pretty efficient in practice.

4 Experiments

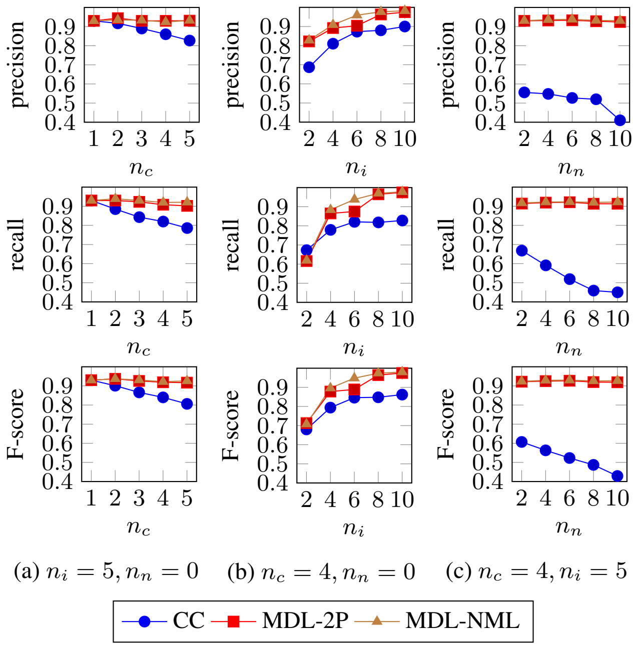
We conduct experiments on both synthetic data and real data. We also present case studies to verify the rationality of our approach. We implemented two versions of MDL based approach: **MDL-2P** using the two-part code, and **MDL-NML** using the normalized maximum likelihood code. We compare them to a clustering-then-conceptualization (**CC** for short) approach. **CC** is an extension of a state-of-the-art single concept conceptualization approach [Song *et al.*, 2011]. In **CC**, we first cluster the input bag of words by K-means with a distance metric defined on their concept distributions in Probase. Then, we find the best single concept for each individual cluster using a naive Bayes model. By comparing to this baseline, we show that the extension of single concept conceptualization in general cannot solve our problem effectively.

4.1 Synthetic data

Data generation. We use synthetic data automatically generated from Probase as the ground truth for evaluation. We use three parameters to guide the generation process: n_c, n_i, and n_n. For each test case, we randomly pick n_c concepts from Probase. We randomly select n_i instances of each concept and use them as the input bag of words. Real data always contains noise. To reflect this, we also randomly select n_n instances other than the instance already selected from the universal instance space of Probase, and add them into the input. We generate $t = 1000$ bags of words for evaluation.

Metric. For each bag of words, we compare the automatically generated concepts to the pre-known concepts. Suppose for the test case i, our approach generates x_i concepts and y_i of them are the pre-known concepts. We quantify the performance using the following metrics: $precision = \frac{\sum_i^t y_i}{\sum_i^t x_i}$, $recall = \frac{\sum_i^t y_i}{t*n_c}$ and $F = \frac{2*precision*recall}{precision+recall}$.

Results. We present the results of *precision, recall* and *F-score* in Figure 1. Since we have three major parameters



(a) $n_i = 5, n_n = 0$ (b) $n_c = 4, n_n = 0$ (c) $n_c = 4, n_i = 5$

Figure 1: Performance on synthetic data

to generate the synthetic data, we fix two of them and vary the remaining one parameter to observe the influence of this parameter on the performance of different solutions. It can be consistently observed that MDL approaches outperform **CC** under almost all the settings. MDL performs especially better when the concept number (n_c) becomes larger; or when more noisy instances (n_n) are introduced. These results suggest that MDL is good at conceptualization with multiple concept and tolerating noises. The results also reveal that **MDL-NML** shows minor superiority over **MDL-2P**.

4.2 Real data

We use two real data sets to evaluate the performance of our approach in real applications:

- **Flickr data** is collected from manually labelled tags in Flickr. Image tags in Flickr are generally redundant and noisy. Conceptual labels refine the tags and help understanding the original tags.
- **Wikipedia data** comes from the results of topic modeling running on the entire Wikipedia corpus. We use LDA[Blei *et al.*, 2003] and extract top words of each topic as the input. Conceptual labeling of the topic words is critical for machine interpretation of the topics.

For the bags of words in real applications, it is difficult to give the ground truth for its best conceptual labels. For example, given tags {`french`, `usa`, `germany`}, either *western country* or *developed country* is an acceptable conceptual label. Hence, we resort to human labeling to evaluate real data. We ask volunteers to manually examine the labeling results and rank their quality with graded scores. The scoring criteria are shown in Table 1. In general, the specific labels that can summarize the meaning of the words own a high score.

Score	Description	Example
3	Perfect	painting, art, portrait, poster → artwork
2	Minor information loss or redundancy	meal, dinner, ceremony, wedding → meal
1	Too vague or specific	tree, plant, flower, agriculture → renewable resource
0	Misleading or unrelated	walkway, swimming pool, vehicle, roof → improvement

Table 1: Criteria for manual evaluation

Algorithm	Flickr data	Wikipedia data
CC	1.04	0.97
MDL-2P	1.9	1.94
MDL-NML	1.98	2.00

Table 2: Evaluation scores on Flickr and Wikipedia data

Results. We manually evaluate the results of 100 test cases randomly selected from each of two data sets. The average score of each approach is shown in Table 2. The MDL based approaches perform consistently better than the competitor in both two real data sets. A closer look at the **CC** approach reveals that **CC** suffers severely from the noises in the real data. Note that Probase is automatically constructed from Web corpus. It also has missing or wrong isA relations, which in general will lead to nonsense labels. However, the MDL based framework tolerates noises much better and achieves better performance.

4.3 Case study

We show the effectiveness of our approach by case studies of Flickr data. We use **MDL-NML** to generate labels since previous evaluation shows that it achieves the best performance. The labeling results are shown in Table 3. The results reveal the following advantages of our approach.

- The conceptual labels are able to summarize the meaning of the tag words.
- The number of concept is adaptable to cover input data. The algorithm can choose an enough number of concepts to cover the different meanings of the given words.
- There is no redundancy between generated concepts.
- The noise filtering is very effective. The noisy words will be automatically selected and excluded.
- The conceptual labels make the tags more interpretable.

Attributes. We also evaluate the effectiveness of incorporating attribute knowledge by case studies. The results are shown in Table 4. The first three cases show that when generating labels for a set of attributes, using isA relations solely often produces nonsense results. But when the attributes data is incorporated, the quality of generated concepts is significantly improved. The last case shows that the addition of attributes relations works well with existing isA relations.

Parameter α. In Section 3.5, we introduce an additional parameter α to adjust the tradeoff between *coverage* and *minimality*. We study the effect of this parameter by varying its

Words	Concepts
handbag$_1$, diaper bag$_1$, hobo bag$_1$, bag$_1$, shoulder bag$_1$, poppy$_2$, flower$_2$, black$_3$, khaki$_3$, white$_3$, coin purse, brand, leather, fashion accessory	bag$_1$, flower$_2$, neutral color$_3$
dance$_1$, nightclub$_1$, disco$_1$, painting$_2$, art$_2$, modern art$_2$, musician, dude	entertainment$_1$, artwork$_2$
furniture$_1$, chair$_1$, shelf$_1$, coffee table$_1$, wood$_2$, box$_2$, wood stain, brown	furniture$_1$, wood$_2$
glasses$_1$, eyewear$_1$, sunglasses$_1$, purple$_2$, brown$_2$, violet$_2$, maroon$_2$, fashion accessory, vision care, product, personal protective equipment	eyewear$_1$, color$_2$
escarpment$_1$, mountain$_1$, mountain range$_1$, ridge$_1$, cliff$_1$, rock$_1$, hill$_1$, outcrop, hill station, trail	terrain feature$_1$

Table 3: Example results of conceptual labeling. Subscripts represent the correspondence between a word and its concept.

Words	$\alpha = 0.6$	$\alpha = 0.5$	$\alpha = 0.4$
shoulder bag, hobo bag, handbag, bag, pink, leather, red, lady, tote bag, coin purse	bag	bag, leather	bag, leather, color
comics, woman, child, person, people, adolescence, bride, girl, sibling, family, daughter	family member	family member	family member, life stage, ornament
hot air balloon, hot air ballooning, vehicle, aircraft, balloon, flight, infant, child, person, hand, abdomen, human body	aircraft	aircraft, body area	aircraft, body area, age group, exciting topic, vehicle
people, community, youth, spring, estate, park, lake, swimming pool, sports, meal, team, powerlifting, snooker, racquet sport	sport	sport, water supply	sport, water supply

Table 5: Results in regard to different settings of parameter α

Words	Labels w/ attr	Labels w/o attr
bride, groom, dress, celebration	wedding	tradition
president, gdp, population	country	macroeconomic variable
crew, manufacturer, captain, weight	ship	capacity
child, infant, toddler, breakfast, lunch	child, meal	child, meal

Table 4: Effectiveness of incorporating attributes data

value on the same input. Several typical cases are shown in Table 5. We can see that the default parameter $\alpha = 0.5$ already works well. Besides, we can increase or decrease α to produce less or more conceptual labels. This gives us the flexibility to adapt to different real requirements about the number of conceptual labels.

5 Conclusion

Explicit semantics are starting to play a more prominent role in text processing. In this paper, we focus on *conceptual labeling*, which aims at generating a minimum set of conceptual labels that best summarize a bag of words. We use a data driven semantic network Probase to find the best concepts. We propose a minimum description length based solution to tradeoff the minimality and coverage constraints on the generated conceptual labels. Extensive experimental results show that our solution is effective in representing the semantics of a bag of words.

References

[Auer *et al.*, 2007] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. *Dbpedia: A nucleus for a web of open data*. 2007.

[Baker *et al.*, 1998] Collin F Baker, Charles J Fillmore, and John B Lowe. The berkeley framenet project. In *ICCL*, pages 86–90, 1998.

[Blei *et al.*, 2003] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[Blei, 2012] David M Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, 2012.

[Bollacker *et al.*, 2008] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, pages 1247–1250, 2008.

[Carlson *et al.*, 2010] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka Jr, and Tom M Mitchell. Toward an architecture for never-ending language learning. In *AAAI*, volume 5, page 3, 2010.

[Etzioni *et al.*, 2004] Oren Etzioni, Michael Cafarella, Doug Downey, Stanley Kok, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S Weld, and Alexander Yates. Web-scale information extraction in knowitall:(preliminary results). In *WWW*, pages 100–110, 2004.

[Hearst, 1992] Marti A Hearst. Automatic acquisition of hyponyms from large text corpora. In *ICCL*, pages 539–545, 1992.

[Lenat, 1995] Douglas B Lenat. Cyc: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38, 1995.

[Miller, 1995] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[Palmer *et al.*, 2010] Martha Palmer, Daniel Gildea, and Nianwen Xue. Semantic role labeling. *Synthesis Lectures on Human Language Technologies*, 3(1):1–103, 2010.

[Rissanen, 1978] Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978.

[Shtar'kov, 1987] Yurii Mikhailovich Shtar'kov. Universal sequential coding of single messages. *Problemy Peredachi Informatsii*, 23(3):3–17, 1987.

[Song *et al.*, 2011] Yangqiu Song, Haixun Wang, Zhongyuan Wang, Hongsong Li, and Weizhu Chen. Short text conceptualization using a probabilistic knowledgebase. In *IJCAI*, pages 2330–2336, 2011.

[Wu *et al.*, 2012] Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Q Zhu. Probase: A probabilistic taxonomy for text understanding. In *SIGMOD*, pages 481–492, 2012.