# Verb Pattern: A Probabilistic Semantic Representation on Verbs

**Wanyun Cui, Xiyou Zhou, Hangyu Lin, Yanghua Xiao***
Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University
wanyuncui1@gmail.com, { xiyouzhou13, 14302010017, shawyh}@fudan.edu.cn

**Haixun Wang**
Facebook, USA
haixun@gmail.com

**Seung-won Hwang**
Yonsei University
seungwonh@yonsei.ac.kr

**Wei Wang**
Shanghai Key Laboratory of Data Science
School of Computer Science, Fudan Uni.
weiwang1@fudan.edu.cn

## Abstract

Verbs are important in semantic understanding of natural language. Traditional verb representations, such as FrameNet, PropBank, VerbNet, focus on verbs' roles. These roles are too coarse to represent verbs' semantics. In this paper, we introduce verb patterns to represent verbs' semantics, such that each pattern corresponds to a single semantic of the verb. First we analyze the principles for verb patterns: generality and specificity. Then we propose a nonparametric model based on description length. Experimental results prove the high effectiveness of verb patterns. We further apply verb patterns to context-aware conceptualization, to show that verb patterns are helpful in semantic-related tasks.

## Introduction

Verb is crucial in sentence understanding (Ferreira and Henderson 1990; Wu and Palmer 1994). A major issue of verb understanding is polysemy (Rappaport Hovav and Levin 1998), which means that *a verb has different semantics or senses when collocating with different objects*. In this paper, we only focus on verbs that collocate with objects. As illustrated in Example 1, most verbs are polysemous. Hence, a good semantic representation of verbs should be aware of their polysemy.

**Example 1** (Verb Polysemy)**.** `eat` *has the following senses:*

- *a. Put food in mouth, chew it and swallow it, such as* `eat apple` *and* `eat hot dog`*.*
- *b. Have a meal, such as* `eat breakfast`*,* `eat lunch`*, and* `eat dinner`*.*
- *c. Idioms, such as* `eat humble pie`*, which means admitting that you are wrong.*

Many typical verb representations, including FrameNet (Baker, Fillmore, and Lowe 1998), PropBank (Kingsbury and Palmer 2002), and VerbNet (Schuler 2005), describe verbs' semantic roles (e.g. ingestor and ingestibles for "eat"). However, semantic roles in general are too coarse to differentiate a verb's fine-grained semantics. A verb in different phrases can have different semantics but similar roles. In Example 1, both "eat"s in "eat breakfast" and "eat apple" have ingestor. But they have different semantics.

The unawareness of verbs' polysemy makes traditional verb representations unable to fully understand the verb in some applications. In sentence `I like eating pitaya`, people directly know "pitaya" is probably one kind of food since eating a food is the most fundamental semantic of "eat". This enables context-aware conceptualization of pitaya to food concept. But by only knowing pitaya's role is the "ingestibles", traditional representations cannot tell if pitaya is food or meal.

**Verb Patterns** We argue that verb patterns (available at http://kw.fudan.edu.cn/verb) can be used to represent more fine-grained semantics of a verb. We design verb patterns based on two word collocation principles proposed in corpus linguistics (Sinclair 1991): *idiom principle* and *open-choice principle*. Following the principles, we designed two types of verb patterns.

- **Conceptualized patterns** According to open-choice principle, a verb can collocate with any objects. Objects have certain concepts, which can be used for semantic representation and sense disambiguation (Wu et al. 2012). This motivates us to *use the objects' concepts to represent the semantics of verbs*. In Example 1, `eat breakfast` and `eat lunch` have similar semantics because both objects have concept `meal`. Thus, we replace the object in the phrase with its concept to form a conceptualized pattern *verb* $\$_C$*concept* (e.g. eat $\$_C$food). Each verb phrase in open-choice principle is assigned to one conceptualized pattern according to the object's concept.
- **Idiom patterns** According to idiom principle, some verb phrases have specific meanings unrelated to the object's concept. We add $\$_I$ before the object to denote the idiom pattern ( i.e. *verb* $\$_I$*object*).

According to the above definitions, we use verb patterns to represent the verb's semantics. Phrases assigned to the same pattern have similar semantics, while those assigned to different patterns have different semantics. By verb pattern-

*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 Municipal Science and Technology Commission foundation key project under No.15JC1400900. Seung-won Hwang was supported by Microsoft.

s, we know the "pitaya" in `I like eating pitaya` is food by mapping "eat pitaya" to "eat $\$_C$food". On the other hand, idiom patterns specify which phrases should not be conceptualized. We list verb phrases from Example 1 and their verb patterns in Table 1. And we will show how context-aware conceptualization benefits from our verb patterns in the application section.

| Verb Phrase | Verb Pattern | Type |
|---|---|---|
| eat apple | eat $\$_C$food | conceptualized |
| eat hot dog | eat $\$_C$food | conceptualized |
| eat breakfast | eat $\$_C$meal | conceptualized |
| eat lunch | eat $\$_C$meal | conceptualized |
| eat dinner | eat $\$_C$meal | conceptualized |
| eat humble pie | eat $\$_I$humble pie | idiom |

Table 1: Verb phrases and their patterns

Thus, our problem is *how to generate conceptualized patterns and idiom patterns for verbs*. We use two public data sets for this purpose: Google Syntactic N-Grams (http://commondatastorage.googleapis.com/books/syntactic-ngrams/index.html) and Probase (Wu et al. 2012). Google Syntactic N-grams contains millions of verb phrases, which allows us to mine rich patterns for verbs. Probase contains rich concepts for instances, which enables the conceptualization for objects. Thus, our problem is given a verb $v$ and a set of its phrases, generating a set of patterns (either conceptualized patterns or idiom patterns) for $v$. However, the pattern generation for verbs is non-trivial. In general, the most critical challenge we face is the trade-off between *generality* and *specificity* of the generated patterns, as explained below.

## Trade-off between Generality and Specificity

We try to answer the question: "*what are good verb patterns to summarize a set of verb phrases*?" This is hard because in general we have multiple candidate verb patterns. Intuitively, good verb patterns should be aware of the *generality* and *specificity*.

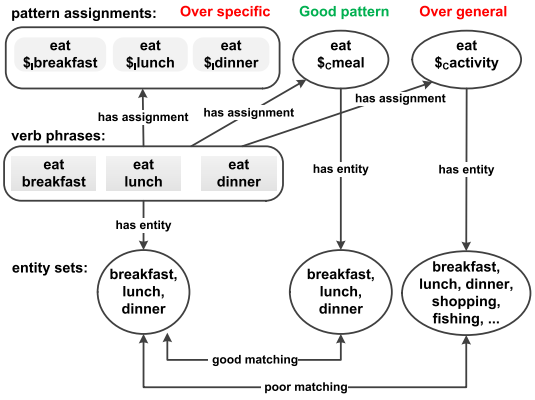


Figure 1: Examples of Pattern Assignments

**Generality** In general, we hope to use fewer patterns to represent the verbs' semantics. Otherwise, the extracted patterns will be trivial. Consider one extreme case where all phrases are considered as idiom phrases. Such idiom patterns obviously make no sense since idioms in general are a minority of the verb phrases.

**Example 2.** *In Fig 1, (*`eat` $\$_C$`meal`*) is obviously better than the three patterns (*`eat` $\$_I$`breakfast + eat` $\$_I$`lunch+ eat` $\$_I$`dinner`*). The former case provides a more general representation.*

**Specificity** On the other hand, we expect the generated patterns are *specific* enough, or the results might be trivial. As shown in Example 3, we can generate the objects into some high-level concepts such as `activity, thing`, and `item`. These conceptualized patterns in general are too vague to characterize a verb's fine-grained semantic.

**Example 3.** *For phrases in Fig1,* `eat` $\$_C$`activity` *is more general than* `eat` $\$_C$`meal`*. As a result, some wrong verb phrases such as* `eat shopping` *or* `each fishing` *can be recognized as a valid instance of phrases for* `eat`*. Instead,* `eat` $\$_C$`meal` *has good specificity. This is because* `breakfast, lunch, dinner` *are three typical instances of* `meal`*, and* `meal` *has few other instances.*

**Contributions** Generality and specificity obviously contradict to each other. How to find a good trade-off between them is the main challenge in this paper. We will use minimum description length (MDL) as the basic framework to reconcile the two objectives. More specifically, our contribution in this paper can be summarized as follows:

- We proposed verb patterns, a novel semantic representation of verbs. We proposed two types of verb patterns: conceptualized patterns and idiom patterns. The verb pattern is polysemy-aware so that we can use it to distinguish different verb semantics.
- We proposed the principles for verb pattern extraction: generality and specificity. We show that the trade-off between them is the main challenge of pattern generation. We further proposed an unsupervised model based on minimum description length to generate verb patterns.
- We conducted extensive experiments. The results verify the effectiveness of our model and algorithm. We presented the applications of verb patterns in context-aware conceptualization. The application justifies the effectiveness of verb patterns to represent verb semantics.

# Problem Model

In this section, we define the problem of extracting patterns for verb phrases. The goal of pattern extraction is to compute: (1) the pattern for each verb phrase; (2) the pattern distribution for each verb. Next, we first give some preliminary definitions. Then we formalize our problem based on minimum description length. The patterns of different verbs are independent from each other. Hence, we only need to focus on each individual verb and its phrases. In the following text, we discuss our solution with respect to a given verb.

## Preliminary Definitions

First, we formalize the definition of *verb phrase*, *verb pattern*, and *pattern assignment*. A verb phrase $p$ is in the form of `verb + object` (e.g. "eat apple"). We denote the

object in $p$ as $o_p$. A verb pattern is either an idiom pattern or a conceptualized pattern. **Idiom Pattern** is in the form of `verb $_I$object` (e.g. eat $\$_I$humble pie). **Conceptualized Pattern** is in the form of `verb $_C$concept` (e.g. eat $\$_C$meal). We denote the concept in a conceptualized pattern $a$ as $c_a$.

**Definition 1** (Pattern Assignment)**.** *A pattern assignment is a function $f : P \to A$ that maps an arbitrary phrase $p$ to its pattern $a$. $f(p) = a$ means the pattern of $p$ is $a$. The assignment has two constraints:*

- *For an idiom pattern* `verb $_I$object`*, only phrase* `verb object` *can map to it.*
- *For a conceptualized pattern* `verb $_C$concept`*, a phrase* `verb object` *can map to it only if the* `object` *belongs to the* `concept` *in Probase (Wu et al. 2012).*

An example of verb phrases, verb patterns, and a valid pattern assignment is shown in Table 1.

We assume the phrase distribution is known (in our experiments, such distribution is derived from Google Syntactic Ngram). So the goal of this paper is to find $f$. With $f$, we can easily compute the pattern distribution $P(A)$ by:

$$P(a) = \sum_p P(a|p)P(p) = \sum_{p \text{ s.t. } f(p)=a} P(p) \tag{1}$$

, where $P(p)$ is the probability to observe phrase $p$ in all phrases of the verb of interest. Note that the second equation holds due to the obvious fact that $P(a|p) = 1$ when $f(p) = a$. $P(p)$ can be directly estimated as the ratio of $p$'s frequency as in Eq 14.

## Model

Next, we formalize our model based on minimum description length. We first discuss our intuition to use Minimum Description Length (MDL) (Barron, Rissanen, and Yu 1998). MDL is based on the idea of data compression. Verb patterns can be regarded as a compressed representation of verb phrases. Intuitively, if the pattern assignment provides a compact description of phrases, it captures the underlying verb semantics well.

Given verb phrases, we seek for the best assignment function $f$ that minimizes the code length of phrases. Let $L(f)$ be the code length derived by $f$. The problem of verb pattern assignment thus can be formalized as below:

**Problem Definition 1** (Pattern Assignment)**.** *Given the phrase distribution $P(P)$, find the pattern assignment $f$, such that $L(f)$ is minimized:*

$$\arg\min_f L(f) \tag{2}$$

We use a two-part encoding schema to encode each phrase. For each phrase $p$, we need to encode its pattern $f(p)$ (let the code length be $l(p, f)$) as well as the $p$ itself given $f(p)$ (let the code length be $r(p, f)$). Thus, we have

$$L(f) = \sum_p P(p)L(p) = \sum_p P(p)[l(p, f) + r(p, f)] \tag{3}$$

Here $L(p)$ is the code length of $p$ and consists of $l(p, f)$ and $r(p, f)$.

**$l(p, f)$: Code Length for Patterns** To encode $p$'s pattern $f(p)$, we need:

$$l(p, f) = -\log P(f(p)) \tag{4}$$

bits, where $P(f(p))$ is computed by Eq 1.

**$r(p, f)$: Code Length for Phrase given Pattern** After knowing its pattern $f(p)$, we use $P_{\mathcal{T}}(p|f(p))$, the probability of $p$ given $f(p)$ to encode $p$. $P_{\mathcal{T}}(p|f(p))$ is computed from Probase (Wu et al. 2012) and is treated as a prior. Thus, we encode $p$ with code length $-\log P_{\mathcal{T}}(p|f(p))$. To compute $P_{\mathcal{T}}(p|f(p))$, we consider two cases:

- Case 1: $f(p)$ is an idiom pattern. Since each idiom pattern has only one phrase, we have $P_{\mathcal{T}}(p|f(p)) = 1$.
- Case 2: $f(p)$ is a conceptualized pattern. In this case, we only need to encode the object $o_p$ given the concept in $f(p)$. We leverage $P_{\mathcal{T}}(o_p|c_{f(p)})$, the probability of object $o_p$ given concept $c_{f(p)}$ (which is given by the isA taxonomy), to encode the phrase. We will give more details about the probability computation in the experimental settings.

Thus, we have

$$\begin{aligned} r(p, f) &= -\log P_{\mathcal{T}}(p|f(p)) \\ &= \begin{cases} -\log P(o_p|c_{f(p)}) & f(p) \text{ is conceptualized} \\ 0 & f(p) \text{ is idiomatic} \end{cases} \end{aligned} \tag{5}$$

**Total Length** We sum up the code length for all phrases to get the total code length $L$ for assignment $f$:

$$\begin{aligned} L(f) &= \sum_p [P(p)l(p, f) + \theta P(p)r(p, f)] \\ &= -\sum_p [P(p)\log P(f(p)) + \theta P(p)\log P_{\mathcal{T}}(p|f(p))] \end{aligned} \tag{6}$$

Note that here we introduce the parameter $\theta$ to control the relative importance of $l(p, f)$ and $r(p, f)$. Next, we will explain that $\theta$ actually reflects the trade-off between the generality and the specificity of the patterns.

## Rationality

Next, we elaborate the rationality of our model by showing how the model reflects principles of verb patterns (i.e. generality and specificity). For simplicity, we define $L_L(f)$ and $L_R(f)$ as below to denote the total code length for patterns and total code length for phrases themselves:

$$L_L(f) = -\sum_p P(p)\log P(f(p)) \tag{7}$$

$$L_R(f) = -\sum_p P(p)\log P_{\mathcal{T}}(p|f(p)) \tag{8}$$

**Generality** We show that by minimizing $L_L(f)$, our model can find general patterns. Let $A$ be all the patterns that $f$ maps to and $P_a$ be the set of each phrase $p$ such that $f(p) = a, a \in A$. Due to Eq 1 and Eq 7, we have:

$$L_L(f) = -\sum_{a\in A}\sum_{p\in P_a} P(p)\log P(a) = -\sum_a P(a)\log P(a) \tag{9}$$

So $L_L(f)$ is the **entropy** of the pattern distribution. Minimizing the entropy favors the assignment that maps phrases to fewer patterns. This satisfies the generality principle.

**Specificity** We show that by minimizing $L_R(f)$, our model finds specific patterns. The inner part in the last equation of Eq 10 actually is the **cross entropy** between $P(P|a)$ and $P_{\mathcal{T}}(P|a)$. Thus $L_R(f)$ has a small value if $P(P|a)$ and $P_{\mathcal{T}}(P|a)$ have similar distributions. This reflects the specificity principle.

$$\begin{aligned} L_R(f) &= -\sum_{a\in A}\sum_{p\in P_a} P(p)\log P_{\mathcal{T}}(p|a) \\ &= -\sum_{a\in A} P(a)\sum_{p\in P_a}\frac{P(p)}{P(a)}\log P_{\mathcal{T}}(p|a) \\ &= -\sum_{a\in A} P(a)\sum_{p\in P_a} P(p|a)\log P_{\mathcal{T}}(p|a) \end{aligned} \tag{10}$$

## Algorithm

In this section, we propose an algorithm based on simulated annealing to solve Problem 1. We also show how we use external knowledge to optimize the idiom patterns.

We adopted a simulated annealing (SA) algorithm to compute the best pattern assignment $f$. The algorithm proceeds as follows. We first pick a random assignment as the initialization (initial temperature). Then, we generate a new assignment and evaluate it. If it is a better assignment, we replace the previous assignment with it; otherwise we accept it with a certain probability (temperature reduction). The generation and replacement step are repeated until no change occurs in the last $\beta$ iterations (termination condition).

**Candidate Assignment Generation** Clearly, the candidate generation is critical for the effectiveness and efficiency of the procedure. Next, we first present a straightforward candidate assignment generation approach. Then, we present an improved solution, which is aware of the typicality of the candidate patterns.

**A Straightforward Generation** The basic unit of $f$ is a single pattern assignment for a phrase. A straightforward aprosch is randomly picking a phrase $p$ and assigning it to a random new pattern $a$. To generate a valid pattern (by Definition 1), we need to ensure either (1) $a$ is the idiom pattern of $p$; or (2) $a$ is a conceptualized pattern and $c_a$ is a hypernym of $o_p$. However, this approach is inefficient since it is slow to reach the optimum state. For a verb, suppose there are $n$ phrases and each of them has $k$ candidate patterns on average. The minimum number of iterations to reach the optimized assignment is $\frac{kn}{2}$ on average, which is unacceptable on big corpus.

**Typicality-aware Generation** We noticed that for a certain phrase, some patterns are better than others due to their high typicality. We illustrate this in Example 4. This motivates us to assign phrases to patterns with higher typicality.

**Example 4.** *Consider* `eat breakfast`, `eat lunch`. `eat` $\$_C$`meal` *is obviously better than* `eat` $\$_C$`activity`. *Since it is more likely for a real human to think up with* `eat` $\$_C$`meal` *than* `eat` $\$_C$`activity` *when he/she sees the phrases. In other words,* `eat` $\$_C$`meal` *is more typical than* `eat` $\$_C$`activity`.

More formally, for a certain phrase $p$, we define $t(p,a)$ to quantify the typicality of pattern $a$ with respect to $p$. If $a$ is an idiom pattern, $t(p,a)$ is set to a constant $\gamma$. If $a$ is a conceptualized pattern, we use the typicality of object $o_p$ with respect to concept $c_a$ to define $t(p,a)$, where $c_a$ is the concept in pattern $a$. Specifically, we have

$$t(p,a) = \begin{cases} \gamma & a \text{ is idiomatic} \\ P_{\mathcal{T}}(o_p|c_a)P_{\mathcal{T}}(c_a|o_p) & a \text{ is conceptualized} \end{cases} \tag{11}$$

, where $P_{\mathcal{T}}(o_p|c_a)$ and $P_{\mathcal{T}}(c_a|o_p)$ can be derived from Probase (Wu et al. 2012) by Eq 15. That is, we consider both the probability from $c_a$ to $o_p$, and that from $o_p$ to $c_a$, to capture their mutual influence.

**Procedure** Now we are ready to present the detailed procedure of our solution:

1. Initialize $f^{(0)}$ by assigning each $p$ to its idiom pattern.
2. Randomly select a new pattern $a$. For each $p$,
$$f^{(i+1)}(p) = \begin{cases} a & t(p,a) > t(p, f^{(i)}(p)) \\ f^{(i)}(p) & otherwise \end{cases} \tag{12}$$
, where $f^{(i)}$ is the assignment in the $i$-th iteration.
3. Accept $f^{(i+1)}$ with probability:
$$p = \begin{cases} 1 & L(f^{(i+1)}) < L(f^{(i)}) \\ e^{(L(f^{(i)}) - L(f^{(i+1)}))/S^A} & L(f^{(i+1)}) \geq L(f^{(i)}) \end{cases} \tag{13}$$
, where $L(f^{(i+1)})$ is the description length for $f^{(i+1)}$, $S$ is the number of steps performed in SA, and $A$ is a constant to control the speed of cooling process.
4. Repeat Step 2 and Step 3, until there is no change in the last $\beta$ iterations.

Step 2 and Step 3 distinguish our algorithm from the generic SA based solution. In Step 2, for each randomly selected pattern $a$, we compute its typicality. If its typicality is larger than that of the currently assigned pattern, we assign the phrase to $a$. In Step 3, we accept the new assignment if its description length is smaller than that in the last round. Otherwise, we accept it with a probability proportional to the exponential of $(L(f^{(i)}) - L(f^{(i+1)}))/S^A$. The rationality is obvious: the larger deviation of $L(f^{(i+1)})$ from $L(f^{(i)})$, the less probable $f^{(i+1)}$ is accepted.

**Complexity Analysis** Suppose there are $n$ phrases. In each iteration, we randomly select a pattern, then we compute the typicality of the pattern for all the $n$ phrases, which costs $O(n)$ time. Next, we compute the description length for $f^{(i+1)}$ by summing up all $n$ phrases' code lengths. This step also costs $O(n)$ time. Suppose our algorithm terminates after $S$ iterations. The entire complexity thus is $O(Sn)$.

**Incorporating Prior Knowledge of Idioms** We notice that many verb idioms can be directly found from external dictionaries. If a verb phrase can be judged as an idiom from dictionaries, it should be directly mapped to its corresponding idiom pattern. Specifically, we first crawled 2868 idiom verb phrases from an online dictionary. Then, in Step 2, when $p$ is one of such idiom phrases, we exclude it from the assignment update procedure.

# Experiments

## Settings

**Verb Phrase Data** The pattern assignment uses the phrase distribution $P(p)$. To do this, we use the "English All" dataset in Google Syntactic N-Grams. The dataset contains counted syntactic ngrams extracted from the English portion of the Google Books corpus. It contains 22,230 different verbs (without stemming), and 147,056 verb phrases. For a fixed verb, we compute the probability of phrase $p$ by:

$$P(p) = \frac{n(p)}{\sum_{p_i} n(p_i)} \tag{14}$$

, where $n(p)$ is the frequency of $p$ in the corpus, and the denominator sums over all phrases of this verb.

**IsA Relationship** We use Probase to compute the probability of an entity given a concept $P_{\mathcal{T}}(e|c)$, as well as the probability of the concept given an entity $P_{\mathcal{T}}(c|e)$:

$$P_{\mathcal{T}}(e|c) = \frac{n(e,c)}{\sum_{e_i} n(e_i,c)} \qquad P_{\mathcal{T}}(c|e) = \frac{n(e,c)}{\sum_{c_i} n(e,c_i)} \tag{15}$$

,where $n(e,c)$ is the frequency that $c$ and $e$ co-occur in Probase.

**Test data** We use two data sets to show our solution can achieve consistent effectiveness on both short text and long text. The short text data set contains 1.6 millions of **tweets** from Twitter (Go, Bhayani, and Huang 2009). The long text data set contains 21,578 **news articles** from Reuters (Apté, Damerau, and Weiss 1994).

## Statistics of Verb Patterns

Now we give an overview of our extracted verb patterns. For all 22,230 verbs, we report the statistics for the top 100 verbs of the highest frequency. After filtering noisy phrases with $n(p) < 5$, each verb has 171 distinct phrases and 97.2 distinct patterns on average. 53% phrases have conceptualized patterns. 47% phrases have idiom patterns. In Table 2, we list 5 typical verbs and their top patterns. The case study verified that (1) our definition of verb pattern reflects verb's polysemy; (2) most verb patterns we found are meaningful.

## Effectiveness

To evaluate the effectiveness of our pattern summarization approach, we report two metrics: (1) ($coverage$) how much of the verb phrases in natural language our solution can find corresponding patterns (2) ($precision$) how much of the phrases and their corresponding patterns are correctly matched? We compute the two metrics by:

$$coverage = \frac{n\_cover}{n\_all} \qquad precision = \frac{n\_correct}{n\_cover} \tag{16}$$

,where $n\_cover$ is the number of phrases in the test data for which our solution finds corresponding patterns, $n\_all$ is the total number of phrases, $n\_correct$ is the number of phrases whose corresponding patterns are correct. To evaluate $precision$, we randomly selected 100 verb phrases from the test data and ask volunteers to label the correctness of their assigned patterns. We regard a phrase-pattern matching is incorrect if it's either too *specific* or too *general* (see examples in Fig 1). For comparison, we also tested two baselines for pattern summarization:

| | |
|---|---|
| verb: **feel** | #phrase: 1355 |
| feel $\$_C$symptom | feel pain (27103), feel chill (4571), ... |
| feel $\$_C$emotion | feel love (5885), feel fear (5844), ... |
| verb: **eat** | #phrase: 1258 |
| eat $\$_C$meal | eat dinner (37660), eat lunch (22695), ... |
| eat $\$_C$food | eat bread (29633), eat meat (29297), ... |
| verb: **beat** | #phrase: 681 |
| beat $\$_I$retreat | beat a retreat (11003) |
| beat $\$_C$instrument | beat drum (4480), beat gong (223), ... |
| verb: **ride** | #phrase: 585 |
| ride $\$_C$vehicle | ride bicycle (4593), ride bike (3862), ... |
| ride $\$_C$animal | ride horse (18993), ride pony (1238), ... |
| verb: **kick** | #phrase: 470 |
| kick $\$_I$ass | kick ass (10861) |
| kick $\$_C$body part | kick leg (703), kick feet (336), ... |

Table 2: Some extracted patterns. The number in brackets is the phrase's frequency in Google Syntactic N-Gram. $\#phrase$ means the number of distinct phrases of the verb.

- **Idiomatic Baseline (IB)** We treat each verb phrase as a idiom.
- **Conceptualized Baseline (CB)** For each phrase, we assign it to a conceptualized pattern. For object $o_p$, we choose the concept with the highest probability, i.e. $\arg\max_c P(c|o_p)$, to construct the pattern.

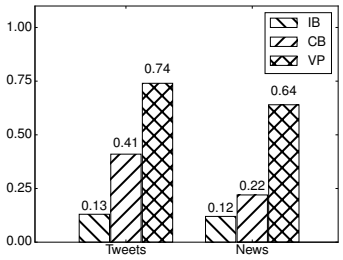


Figure 2: Precision

Verb patterns cover 64.3% and 70% verb phrases in Tweets and News, respectively. Considering the spelling errors or parsing errors in Google N-Gram data, the coverage in general is acceptable. We report the precision of the extracted verb patterns (VP) with the comparisons to baselines in Fig 2. The results show that our approach (VP) has a significant priority over the baselines in terms of precision. The result suggests that both conceptualized patterns and idiom patterns are necessary for the semantic representation of verbs.

# Application: Context-Aware Conceptualization

As suggested in the introduction, we can use verb patterns to improve context-aware conceptualization (i.e. to extract an entity's concept while considering its context). We do this by incorporating the verb patterns into a state-of-the-art entity-based approach (Song et al. 2011).

**Entity-based approach** The approach conceptualizes an entity $e$ by fully employing the mentioned entities in the con-

text. Let $E$ be entities in the context. We denote the probability that $c$ is the concept of $e$ given the context $E$ as $P(c|e, E)$. By assuming all these entities are independent for the given concept, we compute $P(c|e, E)$ by:

$$P(c|e, E) \propto P(e, c)\Pi_{e_i \in E} P(e_i|c) \tag{17}$$

**Our approach** We add the verb in the context as an additional feature to conceptualize $e$ when $e$ is an object of the verb. From verb patterns, we can derive $P(c|v)$, which is the probability to observe the conceptualized pattern with concept $c$ in all phrases of verb $v$. Thus, the probability of $c$ conditioned on $e$ given the context $E$ as well as verb $v$ is $P(c|e, v, E)$. Similar to Eq 17, we compute it by:

$$\begin{aligned} P(c|e, v, E) &= \frac{P(e, v, E|c)P(c)}{P(e, v, E)} \propto P(e, v, E|c)P(c) \\ &= P(e|c)P(v|c)P(E|c)P(c) \\ &= P(e|c)P(c|v)P(v)\Pi_{e_i \in E} P(e_i|c) \\ &\propto P(e|c)P(c|v)\Pi_{e_i \in E} P(e_i|c) \end{aligned} \tag{18}$$

Note that if $v+e$ is observed in Google Syntactic N-Grams, which means that we have already learned its pattern, then we can use these verb patterns to do the conceptualization. That is, if $v + e$ is mapped to a conceptualized pattern, we use the pattern's concept as the conceptualization result. If $v + e$ is an idiom pattern, we stop the conceptualization.

**Settings and Results** For the two datasets used in the experimental section, we use both approaches to conceptualize objects in all verb phrases. Then, we select the concept with the highest probability as the label of the object. We randomly select 100 phrases for which the two approaches generate different labels. For each difference, we manually label if our result is *better* than, *equal* to, or *worse* than the competitor. Results are shown in Fig 3. On both datasets, the precisions are significantly improved after adding verb patterns. This verifies that verb patterns are helpful in semantic understanding tasks.

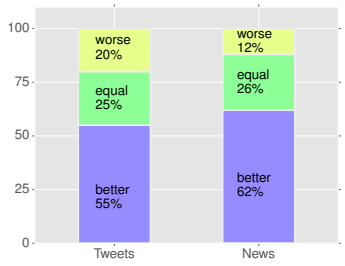


Figure 3: Conceptualization Results

## Related Work

**Traditional Verb Representations** We compare verb patterns with traditional verb representations (Palmer 2009). FrameNet (Baker, Fillmore, and Lowe 1998) is built upon the idea that the meanings of most words can be best understood by semantic frames (Fillmore 1976). Semantic frame is a description of a type of event, relation, or entity and the participants in it. And each semantic frame uses frame elements (FEs) to make simple annotations. PropBank (Kingsbury and Palmer 2002) uses manually labeled predicates and arguments of semantic roles, to capture the precise predicate-argument structure. The *predicates* here are verbs, while *arguments* are other roles of verb. To make PropBank more formalized, the arguments always consist of agent, patient, instrument, starting point and ending point. VerbNet (Schuler 2005) classifies verbs according to their syntax patterns based on Levin classes (Levin 1993). All these verb representations focus on different roles of the verb instead of the semantics of verb. While different verb semantics might have similar roles, the existing representations cannot fully characterize the verb's semantics.

**Conceptualization** One typical application of our work is context-aware conceptualization, which motivates the survey of the conceptualization. Conceptualization determines the most appropriate concept for an entity.Traditional text retrieval based approaches use NER (Tjong Kim Sang and De Meulder 2003) for conceptualization. But NER usually has only a few predefined coarse concepts. Wu et al. built a knowledge base with large-scale lexical information to provide richer IsA relations (Wu et al. 2012). Using IsA relations, context-aware conceptualization (Kim, Wang, and Oh 2013) performs better. Song et al. (Song et al. 2011) proposed a conceptualization mechanism by Naive Bayes. And Wen et al. (Hua et al. 2015) proposed a state-of-the-art model by combining co-occurrence network, IsA network and concept clusters.

**Semantic Composition** We represent verb phrases by verb patterns. while semantic composition works aim to represent the meaning of an arbitrary phrase as a vector or a tree. Vector-space model is widely used to represent the semantic of single word. A straightforward approach to characterize the semantic of a phrase thus is averaging the vectors over all the phrase's words (Xinxiong et al. 2015). But this approach certainly ignores the syntactic relation (Landauer and Dumais 1997) between words. Socher et al. (Socher et al. 2011) represent the syntactic relation by a binary tree, which is fed into a recursive neural network together with the words' vectors. Recently, word2vec (Mikolov et al. 2013a) shows its advantage in single word representation. Mikolov et al. (Mikolov et al. 2013b) further revise it to make word2vec capable for phrase vector. In summary, none of these works uses the idiom phrases of verbs and concept of verb's object to represent the semantics of verbs.

## Conclusion

Verbs' semantics are important in text understanding. In this paper, we proposed verb patterns, which can distinguish different verb semantics. We built a model based on minimum description length to trade-off between generality and specificity of verb patterns. We also proposed a simulated annealing based algorithm to extract verb patterns. We leverage patterns' typicality to accelerate the convergence by pattern-based candidate generation. Experiments justify the high precision and coverage of our extracted patterns. We also presented a successful application of verb patterns into context-aware conceptualization.

## References


Apté, C.; Damerau, F.; and Weiss, S. M. 1994. Automated learning of decision rules for text categorization. *TOIS* 12(3):233–251.

Baker, C. F.; Fillmore, C. J.; and Lowe, J. B. 1998. The berkeley framenet project. In *COLING*, volume 1, 86–90.

Barron, A.; Rissanen, J.; and Yu, B. 1998. The minimum description length principle in coding and modeling. *Information Theory, IEEE Transactions on* 44(6):2743–2760.

Ferreira, F., and Henderson, J. M. 1990. Use of verb information in syntactic parsing: evidence from eye movements and word-by-word self-paced reading. *Journal of Experimental Psychology: Learning, Memory, and Cognition* 16(4):555.

Fillmore, C. J. 1976. Frame semantics and the nature of language. In *Annals of the New York Academy of Sciences: Conference on the origin and development of language and speech*, volume 280, 20–32.

Go, A.; Bhayani, R.; and Huang, L. 2009. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford* 1:12.

Hua, W.; Wang, Z.; Wang, H.; Zheng, K.; and Zhou, X. 2015. Short text understanding through lexical-semantic analysis. In *ICDE*.

Kim, D.; Wang, H.; and Oh, A. 2013. Context-dependent conceptualization. In *IJCAI*, 2654–2661.

Kingsbury, P., and Palmer, M. 2002. From treebank to propbank. In *LREC*.

Landauer, T. K., and Dumais, S. T. 1997. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review* 104(2):211.

Levin, B. 1993. *English verb classes and alternations: A preliminary investigation*. University of Chicago press.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013b. Distributed representations of words and phrases and their compositionality. In *NIPS*, 3111–3119.

Palmer, M. 2009. Semlink: Linking propbank, verbnet and framenet. In *Proceedings of the Generative Lexicon Conference*, 9–15.

Rappaport Hovav, M., and Levin, B. 1998. Building verb meanings. *The projection of arguments: Lexical and compositional factors* 97–134.

Schuler, K. K. 2005. Verbnet: A broad-coverage, comprehensive verb lexicon.

Sinclair, J. 1991. *Corpus, concordance, collocation*. Oxford University Press.

Socher, R.; Huang, E. H.; Pennin, J.; Manning, C. D.; and Ng, A. Y. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *NIPS*, 801–809.

Song, Y.; Wang, H.; Wang, Z.; Li, H.; and Chen, W. 2011. Short text conceptualization using a probabilistic knowledgebase. In *IJCAI*, 2330–2336.

Tjong Kim Sang, E. F., and De Meulder, F. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *CoNLL*, 142–147.

Wu, Z., and Palmer, M. 1994. Verbs semantics and lexical selection. In *ACL*, 133–138.

Wu, W.; Li, H.; Wang, H.; and Zhu, K. Q. 2012. Probase: A probabilistic taxonomy for text understanding. In *SIGMOD*, 481–492.

Xinxiong, C.; Lei, X.; Zhiyuan, L.; Maosong, S.; and Huanbo, L. 2015. Joint learning of character andword embeddings. In *IJCAI*.