Pattern and Distributional Based Hybrid Methods for Semantic Extraction tasks

Guy Kushilevitz
Pattern and Distributional Based Hybrid Methods for Semantic Extraction tasks

Research Thesis

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Computer Science

Guy Kushilevitz

Submitted to the Senate of the Technion — Israel Institute of Technology
Heshvan 5781 Haifa October 2020
This research was carried out under the supervision of Prof. Shaul Markovitch and Prof. Yoav Goldberg, in the Faculty of Computer Science.

Some results in this thesis have been published as a paper by the author and research collaborators in a conference during the course of the author’s MSC research period:


**Acknowledgements**

I would like to thank my advisors, Prof. Yoav Goldberg and Prof. Shaul Markovitch, for their guidance and support throughout this work.

The generous financial help of the Technion is gratefully acknowledged.
## Contents

**List of Figures**

<table>
<thead>
<tr>
<th>Abstract</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Introduction</td>
<td>3</td>
</tr>
<tr>
<td>2 Background, Related Work and Notation</td>
<td>5</td>
</tr>
<tr>
<td>2.1 Pattern-based methods for semantic extraction tasks</td>
<td>5</td>
</tr>
<tr>
<td>2.2 Distribution-based methods for semantic extraction tasks</td>
<td>6</td>
</tr>
<tr>
<td>2.3 Language Modeling</td>
<td>6</td>
</tr>
<tr>
<td>2.4 Notation</td>
<td>7</td>
</tr>
<tr>
<td>3 A Two-Stage Workflow for Semantic Extraction tasks</td>
<td>9</td>
</tr>
<tr>
<td>3.1 First step: finding indicative masked patterns</td>
<td>9</td>
</tr>
<tr>
<td>3.2 Second step: filling the indicative masked patterns</td>
<td>10</td>
</tr>
<tr>
<td>4 Term Set Expansion</td>
<td>11</td>
</tr>
<tr>
<td>4.1 Introduction to the TSE task</td>
<td>11</td>
</tr>
<tr>
<td>4.2 Method</td>
<td>13</td>
</tr>
<tr>
<td>4.2.1 Finding indicative masked-patterns</td>
<td>13</td>
</tr>
<tr>
<td>4.2.2 seed set extension via MLM query</td>
<td>14</td>
</tr>
<tr>
<td>4.2.3 Extended coverage via pattern similarity</td>
<td>14</td>
</tr>
<tr>
<td>4.3 Experiments and Results</td>
<td>15</td>
</tr>
<tr>
<td>4.4 Conclusions</td>
<td>19</td>
</tr>
<tr>
<td>5 Analogy solving</td>
<td>21</td>
</tr>
<tr>
<td>5.1 Introduction to the analogy solving task</td>
<td>21</td>
</tr>
<tr>
<td>5.2 Method</td>
<td>22</td>
</tr>
<tr>
<td>5.2.1 Acquiring indicative masked-patterns</td>
<td>22</td>
</tr>
<tr>
<td>5.2.2 Enhanced pattern picking method</td>
<td>23</td>
</tr>
<tr>
<td>5.2.3 pattern generalization via MLM query</td>
<td>26</td>
</tr>
<tr>
<td>5.3 Experiments and Results</td>
<td>26</td>
</tr>
<tr>
<td>5.4 Conclusions</td>
<td>29</td>
</tr>
</tbody>
</table>
6 Final Remarks

A Additions to Chapter 4
   A.1 Finding candidate terms for TSE ................................................. 33
   A.2 TSE dataset example ................................................................. 33
   A.3 TSE indicative pattern examples ................................................ 35
   A.4 Term Set Expansion Examples .................................................... 38

B Additions to Chapter 5
   B.1 Analogy solving indicative pattern examples ................................. 45
   B.2 Solved analogies ................................................................. 47

Hebrew Abstract i
List of Figures

4.1 TSE dataset statistics ........................................ 16
4.2 TSE Main Results ........................................ 17
4.3 Number of indicative patterns and sentences ................. 18
4.4 Effect of similarity measure’s $k$ on performance ............ 18
4.5 Performance on a subset .................................... 19

5.1 The analogy solving pattern filtering process. ................. 25
5.2 Analogy Dataset statistics ................................... 26
5.3 Main Analogy Solving results ................................ 27
5.4 Tuning the number of sentences parameter for Base method . 28
5.5 Tuning the number of sentences parameter for the Enhanced method . 28
Abstract

Semantic extraction, extracting terms from text according to some property of their meaning, is a core mission in Natural Language Processing (NLP) containing different tasks. These tasks have many stand-alone applications and are also important building blocks for other NLP tasks. Pattern-based methods for semantic extraction are methods in which text patterns that indicate a certain semantic property (e.g. the pattern ```countries such as ___''` that indicates the missing term is a country) are either predefined or searched for and semantic information is extracted by searching for these text patterns in a corpus (e.g. if the text ```countries such as France''` is found in the corpus, we can learn that France is a country). These methods have advantages like transparency but are considered old fashioned and suffer from a lack of performance compared to modern distribution-based methods. Distribution-based extraction methods, based on the famous saying “You shall know a word by the company it keeps” (J.R Firth, 1957), take into account the full word-distribution of a corpus in order to learn about semantic properties that words (or terms) have. For example, based on the linguistic approach that claims that similar words appear in similar contexts, the distribution of a corpus can be used to find words that have similar meanings. Modern distributional tools include Neural Language Models (LM’s), which have revolutionized NLP following the deep learning burst and have boosted performance on many core NLP tasks, including in the field of semantic extraction. One of the main disadvantages of deep-learning-based solutions, and specifically neural LM based solutions, is their lack of transparency. Also, LM based solutions usually require a vast amount of tagged data, even for fine-tuning, which is not easy to come by for some tasks. We present a workflow that combines pattern-based methods and distribution-based methods (specifically, neural LM’s), preserving each of the two’s advantages while avoiding their weaknesses. The gist of the idea is to use the LM to first mine for informative patterns with respect to the specific semantic task we are trying to solve, and then to obtain probable completions for these patterns by generalizing the patterns, in opposed to just searching for them in a corpus like classic pattern-based methods. We demonstrate the use of this workflow for two semantic extraction tasks, the Term Set Expansion (TSE) task and the Analogy Solving task, reaching state-of-the-art results on both tasks while maintaining transparency.
Chapter 1

Introduction

Semantic tasks in Natural Language Processing (NLP) are tasks that focus on words’ or terms’ meanings (as opposed to syntactic tasks, focusing on the structure of a language). Semantic Extraction tasks require extracting terms from a textual corpus such that the meaning of the extracted terms holds some property (which defers between the different semantic tasks). Semantic Extraction uses text and language in order to learn about the relationships that real-world entities, concepts, or objects have. An example of a semantic extraction task might be extract cities in Spain. For this example-task, we will use properties of the natural language in order to find terms in some textual corpus that refer to a city in Spain.

Early techniques, the most famous of which denoted as Hearst patterns [Hea92], proposed a pattern-based approach. For instance, if our task is to extract terms that represent cities in Spain, we can simply search the corpus for terms appearing in the pattern `cities in Spain such as` and extract these terms. These methods, now considered outdated, suffer from a problematic trade-off: while short simple patterns will often lead to noise, long complex (and hence more indicative) patterns will probably not be found enough in the corpus to maintain high recall.

Modern solutions, originating from the famous saying “You shall know a word by the company it keeps” (J.R. Firth, 1957), rely on the distribution of words and terms in a corpus in order to conclude some semantic information about them. Usually, this is done by first composing meaningful embeddings for every word or term and later using these embeddings to draw conclusions about semantic features these words or terms have. These embeddings can be context-less [MSC+13] or, more contemporary, context considering [DCLT19]. Neural models learning these embeddings are trained using a vast amount of training data, and these embedding-originating models are later fine-tuned for downstream tasks (semantic extraction tasks and other NLP tasks), reaching state-of-the-art results on many of them. Having said that, fine-tuning these neural models for downstream tasks usually requires a lot of tagged data, which is not available for all tasks (e.g. the term set expansion task discussed in Chapter 4). Also, these models are considered not transparent: being trained on a lot of data, the
originated embeddings are not meaningful to the human eye, and therefore it is hard to know why a certain decision is made by the model and efficiently deal with mistakes when they appear.

Our proposed workflow combines the pattern-based approach with the distributional-based approach while maintaining each of their advantages. We, like classic pattern-based methods, use indicative patterns: text patterns that are indicative of terms that hold the semantic property we are interested in. After acquiring these patterns, instead of just searching for them in a corpus (like the classic pattern-based methods do) we generalize the acquired patterns (i.e. find words or terms that are probable to fill them) using a distributional method. This allows us to make the patterns longer, more complex, and more indicative compared to classic pattern-based methods because we don’t need to find them in a corpus in order to make them effective. It also utilizes the powerful distributional based neural models for the mission while maintaining transparency: we know which patterns induced the terms we extract and why these patterns were chosen and can act accordingly when mistakes are found.

We adjust and implement our workflow for two semantic tasks, the first of which is Term Set Expansion (TSE). In this task, we get a small set of terms (also known as a seed) and our goal is to expand it into a more complete set of terms that belong to the same semantic class as the seed terms. For example, if our seed is \{“orange”, “red”\} we want to expand it to an (ideally complete) set of colors. The second task we deal with is analogy solving: given two terms that share a semantic relationship, we want to find a term that shares the same relationship with some third input term. For example, for the pair \(Paris, France\) the relationship the two terms share is \texttt{capital_of} and therefore for a third input term \texttt{Jerusalem}, we would like to identify the term \texttt{Israel}. We reach state-of-the-art results on both of these tasks while maintaining the transparent nature of our workflow.

This thesis is structured as follows. Background and related work are discussed in chapter 2. Also, the notation is fixed in this chapter. Our general workflow is described in detail in chapter 3. In chapter 4 (TSE) and chapter 5 (Analogy Solving), we dive into each of the different tasks we implement our workflow on. We describe each task in detail, provide task-specific details on the workflow implementation, and show experiments and results. We share final remarks in chapter 6 and give some supplementary material to the task-specific chapters in appendix A and appendix B.
Chapter 2

Background, Related Work and Notation

In this chapter, we provide a background for our work and discuss past related work. The chapter includes a discussion about pattern-based methods and distributional-based methods for semantic extraction tasks, both building blocks our work leans on. We also survey language modeling work because Masked Language Modeling is a key distributional tool our methods use. Finally, we set the notation that will be used in later chapters of this thesis.

2.1 Pattern-based methods for semantic extraction tasks

The Pattern-Based approach considers specific indicative patterns that signal the desired term or terms, looking for them in a large corpus, and extracting the terms that appear in them. The patterns can be determined manually (Most famous of which are patterns known as “Hearst patterns”, assembled to extract term pairs that are in an “is-a” semantic relationship [Hea92]) or automatically [WC07, GM14].

For example, if we want to extract terms that represent countries, such a pattern can be "countries such as ___.". If this pattern is manually set or automatically chosen we will later search for it in a corpus and if, for example, we find the sentence "countries such as France." we can extract the term “France”.

While well-tailored patterns can be precise and interpretable, a notable shortcoming of pattern-based methods is their lack of coverage, due to the challenge of finding patterns that are specific enough to be accurate yet common enough in a large corpus to be useful. Some past efforts to deal with this problem include using patterns from non-natural language (HTML) [WC07], using part of speech (POS) patterns instead of text patterns [Ril96] and restricting methods to short patterns of 2-4 words to each side of the masked term [GM14]. These restrictions make the methods more vulnerable to noise.
2.2 Distribution-based methods for semantic extraction tasks

Distribution-based methods are used for a wide range of NLP tasks, including semantic extraction. The main idea behind these methods is that one can learn about the properties of words, terms, sentences, or a language by drawing conclusions from the distribution of words in a textual corpus. This idea is supported by linguistics [Fir57, Har68] as well as practical results [MSC+13]. For semantic extraction tasks, this principle is usually implemented by first computing some low dimensional embedding vector for each word or term in the language (which is generated from the word’s distribution in a corpus), and later drawing some semantic conclusions from these embedding vectors. The different distributional semantic extraction methods differ by the way these embeddings are computed [BDVJ03, MSC+13, CWB+11, PSM14, LG14], by considering [DCLT19] or disregarding [MSC+13] context of a word when issuing an embedding, and by the way these embeddings are used in order to draw the semantic conclusions: some draw direct conclusions from the embedding vectors (e.g. to find words similar in meaning one can simply look for words that have similar embedding vectors [MRM18, MPW+18]) and some use these representation vectors as starting points to more complex systems, (e.g. training a classifier that recognizes some properties of these embeddings [Sie15]). State-of-the-art distributional embeddings are context-aware and are learned using neural networks, taking into advantage the Language Modeling task, discussed further in section 2.3.

2.3 Language Modeling

Language Modeling (LM) is a core NLP task, where the goal is to build a model that can predict the probability of a sequence of tokens belonging to a natural language. Having such a model allows us to predict the next token in a sentence by simply looking for the vocabulary token that forms a sentence with the highest predicted probability. First solutions to this problem used N-grams and Markov chains [Mar13, Sha48]: relying on the assumption that one can estimate a probability of a specific token appearing in a sentence seeing only a short history (i.e. a couple of tokens that appear before the token in question) and not the full sentence. Later claims, led by Chomsky [Cho56, CM58] stated that although the N-gram solutions might be good heuristics for the LM problem, they are not capable of being a complete solution for a natural language LM. Never the less, in the absence of a good alternative, these techniques kept being used and improved [BBC+76, CG89, WB91].

The deep learning revolution changed, among other tasks, the commonly used solutions to the LM task. Using Neural Networks (NNs) for this problem allows generalizing from train to test set, as opposed to the N-gram based solutions, because of the ability of NNs to project each word to a vector where similar words have similar vectors. Neural LM
solutions include feed-forward NNs [BDV00], Recurrent NNs [MKB+10], LSTM (Long Short Term Memory) NNs [SSN12] and, most up-to-date, Transformers [VSP+17]. Bert [DCLT19], the state-of-the-art Transformer used in the implementation of this work, implements a Masked Language Model (MLM), a task slightly different from the classic Language Modeling: given a sentence with some tokens marked as masks, it is trained to predict the token most likely to replace the masked token. Modern neural LM’s (and MLM’s) are usually used via transfer learning: their highly valued contextual words representations are used to solve many NLP tasks [DCLT19, HR18, HGJ+19]. Modern LM’s have been shown to contain both semantic [TDP19], syntactic [GoI19, HM19, LDG16] and factual knowledge [PRR+19], and to be great starting points for transfer-learning to new tasks via fine-tuning on (relatively) few examples. These are all very good indicators that neural LM’s contain a great deal of distributional knowledge regarding the corpus they were trained on and are therefore a powerful distributional tool.

2.4 Notation

In this section, we fix some notation that will be used later in this thesis.

- **p**: A pattern, marked as $p$, is a sequence of words or special characters, found in a corpus. For example, a pattern can be "We took Rexy, our pet dog, to the vet!". In this work, the patterns will usually be generated by splitting the corpus where the '.' character is found\(^1\), leading to patterns that are sentences or part of sentences.

- **m**: A masked pattern $m$ is a pattern with (unless explicitly mentioned otherwise) a single masked location (marked as ____), where the mask indicates one or more missing words. For example, a masked pattern can be "We took Rexy, our pet ____ , to the vet!".

- $MLM(m)$ are the word completions (mask replacements) predicted by an MLM for a masked pattern $m$, ranked by their probability. Practically, $MLM(m)$ is an ordered list of size $|MLM\_vocabulary|$ containing all of the MLM’s vocabulary tokens ranked by the probability the MLM predicts each token has to replace the masked location of $m$.

- **t**: a term, marked as $t$, is a sequence of one or more words used to describe an object, an entity, a thing, or a concept.

- $R_{MLM}(t, m)$ is the rank (index) of the term $t$ in the list $MLM(m)$.

---

\(^1\)This is only done for practical implementation reasons, there is no limitation for using multi-sentenced patterns.
Chapter 3

A Two-Stage Workflow for Semantic Extraction tasks

In this chapter, we describe our general semantic extraction workflow. The gist of the idea is to use masked patterns indicative of the term or terms we are looking for (which are determined according to the specific semantic task we are trying to solve), without limiting the patterns in length or complexion. This means the patterns we find will be long and very indicative, but will probably not be found enough in the corpus to be used in the naive way of searching for terms that appear in them. Therefore, instead of just looking for terms that appear in these patterns in a corpus, we will generalize the patterns using a distributional method: we will find terms that are \textit{probable} to fit the masked indicative patterns.

This approach allows us to disregard the complexity of the patterns when picking indicative patterns, and also recruits powerful, knowledge-containing distributional tools to our cause while preserving relative transparency compared to pure distribution-based solutions, since the indicative patterns used are understandable to the human eye.

We operate in two steps. First, we search for indicative masked patterns: patterns that are likely to signal the concept term or class of terms dictated by the task. Then, we use the patterns found and distributional methods to find the term or terms probable to fit the acquired indicative masked patterns.

3.1 First step: finding indicative masked patterns

An indicative masked pattern is a masked pattern \( m \) such that, with high probability, the term or terms we are looking to extract will make good mask replacements, while terms that are not the terms we desire will make bad replacements.

For example, if our semantic task is to find terms that represent countries, the masked pattern \( m_1 = \text{“The capital of \\_\\_”} \) is a good indicative masked pattern because terms probable to fill the mask in \( m_1 \) have a capital and are therefore likely to be
countries. Also, terms that are not countries are not likely to fit \( m_1 \), making it a good indicative masked pattern for the semantic class of countries. The masked pattern \( m_2 = \text{“You visited ____ last summer.”} \) is not a good indicative pattern for the country semantic class because although a term representing a country can definitely fill \( m_2 \), other terms (e.g. terms representing a city) are likely to fit \( m_2 \) as well.

The output of the first step of the workflow, which will also be the input to the second step, is a group of masked indicative patterns indicative of the concept term or terms we eventually want to find. The way to acquire these patterns is task-specific; for the TSE task our implementation of this stage is reported in section 4.2.1 and for the analogy solving task it is discussed in section 5.2.1.

### 3.2 Second step: filling the indicative masked patterns

From the definition of a masked indicative pattern, as explained in section 3.1, and assuming the patterns are well selected in the first step, terms probable to fill the indicative masked patterns should be the terms we want to extract. Therefore, After acquiring indicative masked patterns, we simply want to find a term or terms that are probable to fit them. A naive way of doing so would be to search a corpus for appearances of the indicative masked patterns and extract terms that appear in them. For short masked indicative patterns, this naive approach will usually lead to unwanted terms being extracted. Long and complex indicative masked patterns, like the ones hopefully acquired in the first step (since we do not limit length or complexity), will probably not be found in the corpus and therefore would not be helpful with this naive approach. This calls for a different solution: instead of just searching for the acquired indicative masked patterns in a corpus, we will use distributional techniques to find terms that are probable to appear in these patterns.

Although this can potentially be done with various distributional tools, we choose to use Neural MLM’s as our distributional tool for the task of generalizing the patterns. We do so because, besides their proven ability to perform well on various NLP tasks, these models are trained precisely for the task we desire; finding words or terms that are probable to fit a masked pattern. Neural MLM’s have very good generalization ability, meaning they can find terms probable to fit a masked pattern even if they never saw this exact masked pattern during training.

Task-specific implementation details for this step can be found at section 4.2.2 and section 4.2.3 for the TSE task and at section 5.2.3 for the analogy solving task.
Chapter 4

Term Set Expansion

In this chapter, we dive into the Term Set Expansion (TSE) task. We describe it, give our workflow’s implementation\(^1\) on this task and report experiments and results.

4.1 Introduction to the TSE task

**Term Set expansion** (TSE) is the task of expanding a small seed set of terms into a larger (ideally complete) set of terms that belong to the same semantic category. For example, the seed set \{“orange”, “apple”\} should expand into a set of *fruits*, while \{“orange”, “blue”\} into a set of *colors*, and \{“apple”, “google”\} into a set of *tech companies*. Beyond being of great practical utility, the TSE task is a challenging instance of a *generalization from few examples* problem. Solving TSE requires the algorithm to: (1) identify the desired concept class based on a few examples; and (2) identify additional members of the class.

We present an effective TSE method which is based on querying large, pre-trained masked language models (MLMs). Pre-trained language models (LMs) have been shown to contain semantic \([\text{TDP19}], \text{syntactic} [\text{Gol19}, \text{HM19}, \text{LDG16}] \text{and} \text{factual knowledge} [\text{PRR}^+19], \text{and} \text{to be great starting points for transfer-learning to new tasks via fine-tuning on few examples. However, the TSE seed sets are too small for fine-tuning, calling for a different approach. Our method uses the MLMs directly for the task they were trained for—language-modeling—by issuing word-completion queries and operating on the returned word distributions.}^2\)

**Previous solutions** to the TSE problem (also called semantic class induction) can be roughly categorized into *distributional* and *pattern-based* approaches \([\text{SZYW10}]. \text{Our method can be seen as a combination of the two.}

The distributional approach to TSE \([\text{Hin90}, \text{PL02}, \text{PCB}^+09, \text{MPW}^+18, \text{MRM18}]\) operates under the hypothesis that similar words appear in similar contexts \([\text{Har68}].\)

\(^{1}\text{Code and data are available at https://github.com/guykush/TermSetExpansion-MPB/}.

\(^{2}\text{See [AG18] for a method that uses MLM word completions for word-sense induction.}\)
These methods represent each term in the vocabulary as an embedding vector that summarizes all the contexts the term appears in in a large corpus, and then look for terms with vectors that are similar to those of the seed term. The methods differ in their context definitions and in their way of computing similarities. A shortcoming of these methods is that they consider all occurrences of a term in the corpus when calculating its representation, including many contexts that are irrelevant to the concept at hand due to polysemy, noise in the corpus, or non-informative contexts.\(^3\)

In contrast, the pattern-based approach considers specific indicative patterns that signal the desired concept, looking for them in a large corpus, and extracting the terms that appear in them. Patterns can be binary [Hea92, OOT06, ZZSW09] (“such as X or Y”), indicating that both X and Y belong to the same class, or unary [GM14, WC07] (“fruits such as X”, “First I painted the wall red, but then I repainted it X”), suggesting that X belongs to a certain category (fruit, color). The patterns can be determined manually [Hea92] or automatically [WC07, GM14]. While well-tailored patterns can be precise and interpretable, a notable shortcoming of pattern-based methods is their lack of coverage, due to the challenge of finding patterns that are specific enough to be accurate yet common enough in a large corpus to be useful. [WC07] use patterns from non-natural language (HTML) while [GM14] restrict themselves to short patterns of 2-4 words to each side of the masked term.

**Our method.** By using MLMs, we combine the power of the pattern-based and the distributional approaches: like the patterns-based approaches, we consider only specific, indicative corpus locations (retaining specificity and transparency). We then use the distributional nature of the neural LM to generalize across patterns and corpus locations.

We use masked patterns that are indicative of the semantic class in questions. For example, \''We took Rexy, our pet _____, to the vet.'\' is an indicative masked pattern for the house animals semantic class. Given an initial set of seed terms, we first search the corpus for masked patterns indicative of members of the set (4.2.1). Intuitively, an indicative masked pattern is a corpus location which is considered by an MLM to be a good fit for all seed members. Once we identified indicative patterns, we extend the set to terms that can appear in similar patterns. We propose two methods for doing this. The first method (4.2.2) queries an MLM for completions. While effective, this method restricts the expanded set to the MLM’s vocabulary. The second method (4.2.3) uses the MLM to define a similarity metric over patterns and searches the corpus for terms that appear in patterns that are similar to the indicative ones. To summarize, we embrace the pattern-based approach, while using distributional similarity for identifying good patterns as well as for generalizing across patterns.

---

\(^3\)The work of [MRM18] is unique in this regard by considering only a subset of the contexts that are relevant for the expansion, as determined from the seed set.
4.2 Method

**Task formulation** we are given a seed set $S$ of $k$ terms, $S = t_1, ..., t_k$, that come from a larger (and unknown) gold set $S_g$. Our goal is to return $S_g$. Practically, our (and other) algorithms return a ranked list of terms rather than a fixed set. The evaluation is then performed over the ranking: ideally, all terms in $S_g$ will be ranked above all terms not in $S_g$.

We operate in stages. First, we search the corpus for $\ell$ indicative masked patterns $m_1, ..., m_\ell$, that are likely to signal the concept class in $S_g$ with high probability. Then, we use the patterns to extend the set.

4.2.1 Finding indicative masked-patterns

We look for masked patterns such that, with high probability, instances of the desired semantic class will make good mask replacements, while instances of other classes will make bad replacements. For example, “The capital of ____” is a good masked pattern for the “countries” class. We collect $L$ masked pattern candidates for each seed term $t_j$ by querying a corpus for sentences that contain the term and replacing the term position with a mask. We then score each of the $kL$ resulting pattern candidate $m_i$ and take the $\ell$-best ones.

Intuitively, we seek a diverse set of patterns in which all seed terms are ranked high (i.e., have a low-rank index) in the MLM’s prediction: we look for patterns whose worst-fitting seed term is still high on the list of replacement terms. Formally, we use $MLM(m)$ (the list of word completions, or mask replacements, predicted by the MLM for the masked pattern $m$ ranked by their probability) and $R_{MLM}(t, m)$ (the rank, or index, of the term $t$ in $MLM(m)$) to define a score for every candidate masked pattern. The score of a pattern is then the maximal rank of any of the seed terms:

$$s(m_i) = \max_{t_j \in S} R_{MLM}(t_j, m_i)$$

We then sort the patterns by $s(m_i)$ and take the patterns with minimal values. This min-over-max formulation ensures that the patterns are a good fit for all seed terms.

To achieve the diversity objective, we use the following heuristic: after sorting all candidate patterns $m_i$ by $s(m_i)$, rather than taking the first $\ell$ items we go over the sorted list in order and keep a pattern only if it differs by at least 50% of its tokens from all patterns already kept. We do this until collecting $\ell$ patterns.

Examples for indicative masked patterns found using the described method can be found at appendix A.3.

---

4 In this work we focus on small values of $k$. Our experiments use $k = 3$ seed terms.
5 We assume the seed terms are a part of the MLM’s vocabulary.
6 Contrast this to a min-over-average formulation, which may score very well on some seed terms but badly on others.
4.2.2 seed set extension via MLM query

Having identified indicative masked patterns, we now turn to suggest terms for expanding the seed set. Each indicative masked pattern $m_i$ naturally provides a ranked list of candidate terms $MLM(m_i) = t_1, ..., t_{|V|}$, where $V$ is the MLM’s vocabulary and each term $t_j$ is scored by its pattern-conditional probability. We combine the term scores from all chosen indicative patterns using a product of experts approach, scoring each term by the product of probabilities (sum of log probabilities) assigned to it by each context. Let $p_{MLM}(t|m_i)$ be the probability assigned to vocabulary term $t$ for replacing the mask in the masked pattern $m_i$. The term score is:

$$score(t) = \sum_{i=1}^{\ell} c_i \log p_{MLM}(t|m_i)$$

(4.2)

where $c_i = \frac{\text{maxRank}(m_i)^{-1}}{\sum_{j=1}^{\text{maxRank}(m_i)^{-1}}}$ is a weighing factor for every indicative masked pattern $m_i$, giving more weight to “tighter” indicative masked patterns.

This method is fast and effective, requiring only $\ell$ queries to the MLM. However, it assumes that all the desired terms from $S_g$ appear as vocabulary items in the MLM. This assumption often does not hold in practice: first, for efficiency reasons, pre-trained MLM vocabularies are often small ($\sim 50k$ items), precluding rare words. Second, many terms of interest are multi-word units, that do not appear as single items in the MLM’s vocabulary.

4.2.3 Extended coverage via pattern similarity

We seek a term set expansion method that will utilize the power of the pre-trained LM, without being restricted by its vocabulary: we would like to identify rare words, out-of-domain words, and multi-word units.

Our solution is to generalize the indicative patterns. Rather than looking for terms that match the patterns, we instead search a large corpus for patterns which are similar to the indicative ones, and collect the terms that appear within them. Following the distributional hypothesis, these terms should be of the desired concept class.

By looking at patterns that surround corpus locations, we are no longer restricted by the MLM vocabulary to single-token terms.

However, considering all corpus locations as candidate patterns is prohibitively expensive. Instead, we take a ranking approach and restrict ourselves only to corpus locations that correspond to occurrences of candidate terms returned by a high-recall algorithm.

We use the LM to define a similarity measure between two masked patterns that

---

7For example, one that is based simple distributional similarity to the seed terms. In this work we use the nearest neighbours returned by the sense2vec model [TML15], as implemented in [https://spacy.io/universe/project/sense2vec](https://spacy.io/universe/project/sense2vec). More details can be found at appendix A.1.
aims to capture our desired notion of similarity: masked patterns are similar if they are likely to be filled by the same terms. Let \( \text{top}_q(\text{MLM}(m_i)) \) be the \( q \) terms most likely to replace the mask in the pattern \( m_i \) as predicted by the MLM. We define the similarity between two patterns as the fraction of shared terms in their top \( q \) predictions (\( q \) being a hyperparameter):

\[
\text{sim}(m_i, m_j) = \frac{|\text{top}_q(\text{MLM}(m_i)) \cap \text{top}_q(\text{MLM}(m_j))|}{q}
\]

For a candidate term \( t \), let \( \text{pats}(t) = m^t_1, ..., m^t_n \) be the set of patterns derived from it: sentences that contain \( t \), where \( t \) is replaced with a mask. Note that \( t \) can be an arbitrary word or word sequence. We wish to find terms for which the similarity between \( \text{pats}(t) \) and the indicative patterns is high. However, since words have different senses, it is sufficient for only some patterns in \( \text{pats}(t) \) to be similar to patterns in \( m_1, ..., m_\ell \). We score a term \( t \) as:

\[
\text{score}(t) = \sum_{i=1}^{\ell} c_i \max_{m \in \text{pats}(t)} \text{sim}(m_i, m)
\]

(4.3)

where \( c_i \) is the pattern weighing factor from equation (2). As \( \sum_{i=1}^{\ell} c_i = 1 \), the term score \( \text{score}(t) \) for every term \( t \) is \( \in [0, 1] \).

### 4.3 Experiments and Results

We refer to the method in Section (4.2.2) as MPB1 and the method in section (4.2.3) as MPB2.

**Setup.** In our experiments, we use BERT [DCLT19] as the MLM and English Wikipedia as the corpus. Following previous TSE work (e.g. [MRM18]), we measure performance using MAP (using MAP\(_{70}\) for the open set). For each method, we report the average MAP over several runs (exact number mentioned under each figure), each with a different random seed set of size 3. Based on preliminary experiments, for MPB1 we use \( \ell = 160 \) and \( L = 2000/k \) and for MPB2 we use \( \ell = 20 \) and \( L = 2000/k \).\(^8\) When comparing different systems (i.e, in figure 4.2), each system sees the same random seed sets as the others. For smaller sets, we expand to a set of size 200, while for the Countries and Capitals sets, which have expected sizes of \( > 100 \), we expand to 350 items.

**Dataset.** Automatic TSE evaluation is challenging. A good TSE evaluation set should be complete (contain all terms in the semantic class), clean (not contain other terms) and comprehensive (contain all different synonyms for all terms). These are hard to come by. Indeed, previous work either used a small number of sets or used some automatic set acquiring method which is commonly not complete. We curated a dataset with 7 closed, well-defined sets, which we make publicly available. The sets are National football league teams (NFL, size:32), Major league baseball teams (MLB, 30), US states (States, 50), Countries (Cntrs, 195), European countries (Euro, 44)\(^8\)see Additional experiments for a justification of these parameter choices.
<table>
<thead>
<tr>
<th>Set</th>
<th>Number of terms</th>
<th>Average number of synonyms</th>
</tr>
</thead>
<tbody>
<tr>
<td>NFL</td>
<td>32</td>
<td>4.906</td>
</tr>
<tr>
<td>MLB</td>
<td>30</td>
<td>5.333</td>
</tr>
<tr>
<td>States</td>
<td>50</td>
<td>3.000</td>
</tr>
<tr>
<td>Cntrs</td>
<td>196</td>
<td>2.010</td>
</tr>
<tr>
<td>Euro</td>
<td>44</td>
<td>2.272</td>
</tr>
<tr>
<td>Caps</td>
<td>191</td>
<td>1.10</td>
</tr>
<tr>
<td>Pres</td>
<td>44</td>
<td>5.068</td>
</tr>
<tr>
<td>Genre</td>
<td>656</td>
<td>1</td>
</tr>
</tbody>
</table>

Figure 4.1: TSE dataset statistics

Capital cities (Caps, 195) and Presidents of the USA (Pres, 44). We also provide on
one open class set: Music Genres (Genre). This set is created by manually verifying
the items in the union of the output of all the different algorithms. This set contains
around 600 unique items. An example class from our generated dataset is shown at
appendix A.2. Dataset statistics are presented at fig. 4.1. The full dataset is available
at https://github.com/guykush/TermSetExpansion-MPB/.

**Compared Methods.** We compare our methods, MPB1 (MLM-pattern-based) (Sec-
tion 4.2.2) and MPB2⁹ (Section 4.2.3), to two state-of-the-art systems: setExpander¹⁰
(SE) [MPW⁺18], and category builder (CB) [MRM18]. We also compare to two base-
lines: The first, BB (basic-BERT), is a baseline for MPB1. This is a BERT-based
baseline that uses the MPB1 method on patterns derived from sentences that include
seed terms, without the selection method described in Section 4.2.1. The second, s2v,
is a baseline for MPB2. This is a basic distributional method that uses sense2vec
[TML15] representations,¹¹ which is also our candidate acquisition method for MPB2
(appendix A.1). As MPB2 relies on external candidate generation, we also report on
the oracle case MPB2+O where we expand the s2v-generated candidate list to include
all the members of the class.

---

⁹We follow [MRM18] and limit MPB2 to 200,000 most frequent terms. MPB2 can work with any
number of terms and is limited only by the candidate supplying method (in this implementation
sense2vec which has ∼3,400,000 terms).

¹⁰We use the non-grouping release version because it reaches better results on our dataset than the
grouping one.

¹¹https://explosion.ai/demos/sense2vec
Main Results. Our main results are reported in figure 4.2. Our first method, MPB1, achieves the best scores on two of the three sets suitable for its limitations (where all or most of the set’s terms are in the LM’s vocabulary), and second-best results on the third. MPB2 outperforms all other methods on 5 out of 7 closed sets when assuming gold-standard candidates (MPB2+O), and even when considering the missing candidates it outperforms other expanders on 4 out of 7 closed sets, averaging the best MAP score on all sets. While other methods tend to stand out in either closed sets (CB) or the open set (SE), MPB2 shows good performance on both kinds of sets. The results also suggest that a better candidate-acquiring method may lead to even better performance. Expansion examples for our methods can be found at appendix A.4.

Additional experiments. How many sentences should we query when searching for indicative patterns, and how many patterns should we retain? Figure 4.3 shows a grid of these parameters. We use the NFL set for this experiment, as terms in this set all have more than one meaning, and for most, the common usage is not the one that belongs to the NFL set (e.g. “jets”, “dolphins”). Therefore, this set should give a pessimistic estimation for the number of sentences we need to extract in order to find quality indicative patterns. Results imply that \(~2000\) appearances of seed terms are sufficient, and that good results can be obtained also with fewer instances. This shows that—beyond the data used to train the initial MLM—we do not require a large corpus to achieve good results, suggesting applicability also in new domains.

---

**Table 4.2: TSE Main Results**

<table>
<thead>
<tr>
<th>Method</th>
<th>NFL</th>
<th>MLB</th>
<th>Pres</th>
<th>States</th>
<th>Cntrs</th>
<th>Euro</th>
<th>Caps</th>
<th>Genre</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>SE</td>
<td>.54</td>
<td>.45</td>
<td>.33</td>
<td>.55</td>
<td>.55</td>
<td>.61</td>
<td>.14</td>
<td>.99</td>
<td>.52</td>
</tr>
<tr>
<td>CB</td>
<td>.98</td>
<td>.97</td>
<td>.70</td>
<td>.93</td>
<td>.74</td>
<td>.46</td>
<td>.21</td>
<td>.67</td>
<td>.71</td>
</tr>
<tr>
<td>BB</td>
<td>.91</td>
<td>.92*</td>
<td>.52**</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>.78†</td>
</tr>
<tr>
<td>MPB1</td>
<td>.98</td>
<td>.99*</td>
<td>.63**</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>.87†</td>
</tr>
<tr>
<td>S2V</td>
<td>.95</td>
<td>.80</td>
<td>.18</td>
<td>.94</td>
<td>.71</td>
<td>.78</td>
<td>.21</td>
<td>.90</td>
<td>.68</td>
</tr>
<tr>
<td>MPB2</td>
<td>.95</td>
<td>.82</td>
<td>.37</td>
<td>.98</td>
<td>.76</td>
<td>.79</td>
<td>.27</td>
<td>.98</td>
<td>.74</td>
</tr>
<tr>
<td>MPB2+O</td>
<td>.95</td>
<td>.90</td>
<td>.88</td>
<td>.98</td>
<td>.91</td>
<td>.81</td>
<td>.80</td>
<td>NA*</td>
<td>.89†</td>
</tr>
</tbody>
</table>

Average MAP scores over 3 random seeds of size 3. */**: excluding 2 or 3 OOV terms. **NA**: Not applicable, because sets contain many OOV terms. **NA**: Not applicable for oracle setting, because gold standard candidates not available for open sets. †: Average value over applicable sets only.

---

\(^{12}\)MPB1’s relatively poor performance on the president’s set can be a result of the basic terms MPB1 considers. MPB1 ranks only terms which are in the LM’s vocabulary, which means that while other expanders can rank terms like “President George W. Bush”, MPB1 will consider terms like “bush”, which are harder to ascribe to the presidents set. While this is true for all sets, it seems to be more significant for a set containing person names.

\(^{13}\)SE does not rank the seed terms, as opposed to other methods. For fairness, we add them in the beginning of the returned list before computing the MAP score.
## Patterns and Sentences

<table>
<thead>
<tr>
<th># patterns</th>
<th>20</th>
<th>100</th>
<th>300</th>
<th>1000</th>
<th>2000</th>
<th>4000</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.794</td>
<td>.729</td>
<td>.704</td>
<td>.843</td>
<td>.939</td>
<td>.939</td>
</tr>
<tr>
<td>5</td>
<td>.834</td>
<td>.938</td>
<td>.960</td>
<td>.969</td>
<td>.981</td>
<td>.964</td>
</tr>
<tr>
<td>10</td>
<td>.839</td>
<td>.938</td>
<td>.974</td>
<td>.978</td>
<td>.990</td>
<td>.975</td>
</tr>
<tr>
<td>20</td>
<td>.838</td>
<td>.932</td>
<td>.972</td>
<td>.987</td>
<td>.990</td>
<td>.978</td>
</tr>
<tr>
<td>40</td>
<td>NA</td>
<td>.916</td>
<td>.962</td>
<td>.993</td>
<td>.993</td>
<td>.989</td>
</tr>
<tr>
<td>80</td>
<td>NA</td>
<td>.913</td>
<td>.954</td>
<td>.992</td>
<td>.996</td>
<td>.993</td>
</tr>
<tr>
<td>160</td>
<td>NA</td>
<td>NA</td>
<td>.949</td>
<td>.985</td>
<td><strong>.998</strong></td>
<td>.997</td>
</tr>
<tr>
<td>600</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>.981</td>
<td>.994</td>
<td>.993</td>
</tr>
</tbody>
</table>

Figure 4.3: Number of indicative patterns and sentences

Number of indicative patterns used and number of candidate seed-term containing sentences used for selecting these indicative patterns. Set is the NFL team set, method is MPB1. Every value is an avg MAP on 5 seeds (chosen randomly, fixed for all values of the parameters shown in the table) of size 3. NA: Number of indicative patterns can not be bigger than number of candidate seed-term containing sentences.

While for MPB1 there are no prominent downsides in using a large number of indicative patterns, for MPB2 doing so will force us to use a large number of occurrences of the candidate terms also. This will (1) be costly run-time wise and (2) many occurrences of rare terms might not always be available. Therefore, we choose different parameters for MPB1 and MPB2. While in both we will use 2000 sentences to search for these indicative patterns ($L = 2000/k$), for MPB1 we will use 160 indicative patterns ($\ell = 160$) and for MPB2 we will use only 20 of them ($\ell = 20$).

How sensitive is the algorithm to the choice of $k$ when computing the pattern similarity? Figure 4.4 shows that the similarity measure is effective for various $k$ values, with max performance at ~50.

<table>
<thead>
<tr>
<th>Set</th>
<th>$k=1$</th>
<th>$k=5$</th>
<th>$k=50$</th>
<th>$k=300$</th>
<th>$k=700$</th>
<th>$k=3000$</th>
</tr>
</thead>
<tbody>
<tr>
<td>States</td>
<td>.693</td>
<td>.848</td>
<td>.986</td>
<td>.965</td>
<td>.972</td>
<td>.975</td>
</tr>
<tr>
<td>NFL</td>
<td>.876</td>
<td>.939</td>
<td>.938</td>
<td>.919</td>
<td>.921</td>
<td>.916</td>
</tr>
</tbody>
</table>

Figure 4.4: Effect of similarity measure’s $k$ on performance

Using MPB2 on a single random seed from each set.

18
Finally, how do the different methods behave in a case where the seed terms are a part of a subset? Figure 4.5 shows a case where seed terms are European countries. Ideally, we would like top results to be European countries, later results to be non-European countries and then unrelated terms. MPB2+O achieves the best MAP scores on both the set and the subset. In the subset case, even when not provided with all oracle terms, MPB2 is better than all other expanders. While other expanders tend to reach stronger results on either the set or the subset, MPB2+O achieves similar scores on both.

### 4.4 Conclusions

We introduce an LM-based TSE method, reaching state-of-the-art results. The method uses the power of LM predictions to locate indicative patterns for the concept class indicated by the seed terms, and then to generalize these patterns to other corpus locations. Beyond strong TSE results, our method demonstrates a novel use of pre-trained MLMs, using their predictions directly rather than relying on their states for fine-tuning.
Chapter 5

Analogy solving

In this chapter, we dive into the analogy solving task. We describe the task, describe our workflow’s implementation on it, and report experiments and results. Our method outperforms state-of-the-art analogy solving algorithms while preserving the transparent nature of our workflow (chapter 3) that allows interoperability and understanding of the decisions our method makes.

5.1 Introduction to the analogy solving task

Analogy Solving is a task where given an input 2-tuple of terms (denoted \((a : a^*)\) in this work) that are in a certain semantic relationship with each other, and a third term \(b\), we would like to extract a term \(b^*\) such that the third term \(b\) and the extracted term share the same semantic relationship as the input 2-tuple of terms \((a : a^*)\). For example, if the input 2-tuple is \("Paris" : "France"\) the two terms are in the following semantic relationship: the first term is the capital of the second term. We will mark this relationship as \(\text{capital\_of}\) and denote that \(\text{capital\_of}(t, t^*)\) holds if the term \(t\) represents the capital of \(t^*\). Note that \(\text{capital\_of}("Paris", "France")\) holds. Now, if our third input term \(b\) is "Berlin", we would like to find a term \(b^*\) such that \(\text{capital\_of}("Berlin", b^*)\) holds. The term "Germany" would be the correct answer in this case. Analogy solving is usually done in a pure-distributional way: originating from \([\text{MSC}^{+13}], [\text{MCCD}13]\), embeddings are calculated for each term according to the distribution of some corpus, and the relationship between two terms is treated as the difference between the embedding vectors. Marking the embedding vector calculated for a term \(t\) as \(V_t\), the relationship vector for the input 2-tuple \((a : a^*)\) can simply be calculated as \(V_a - V_{a^*}\). Then, to get the estimated embedding vector of the wanted term \(b^*\) this relationship vector is added to the vector of the third term \(b\): \(V_b + (V_a - V_{a^*})\). Finally, the vocabulary term that has the embedding vector which is the closest to this estimated vector \((V_b + (V_a - V_{a^*}))\) is searched for and returned as the predicted \(b^*\).
5.2 Method

Like other implementations of our workflow (chapter 3), we will operate in two stages. First, we will search for indicative patterns, and then we will fill them using the MLM.

Task formulation We are given a 2-tuple \((a, a^*)\) of input terms that represent objects, concepts, or entities that are in some semantic relationship \(r^*\). We will denote a semantic relationship \(r(t, t^*)\) is true only if the terms \(t\) and \(t^*\) represent objects, concepts, or entities that share this certain semantic relationship. Given a question term \(b\), we would like to find an output term \(b^*\) such that for some unknown semantic relationship \(r^*\), \(r^*(a, a^*)\) is true and \(r^*(b, b^*)\) is also true.

5.2.1 Acquiring indicative masked-patterns

For this task, we look for indicative masked patterns such that, with high probability, the term probable to fill the masked pattern will be a term that holds the desired relationship with the question term \(b\). We seek a diverse set of patterns where the first term in the relationship dictates the second one. To acquire masked patterns that are relationship-indicative we will operate in the following stages:

- Collect \(L\) pattern candidates.
- Pick a subgroup of the collected patterns by filtering out patterns that are not relationship-indicative. This is the crucial step of acquiring the indicative masked patterns. We describe a basic filter in this section and an enhanced one in section 5.2.2
- Transform the picked patterns to masked-patterns that are indicative of the specific term we are looking for \((b^*)\).

We will achieve the diverse goal with the following heuristic: after filtering out the non-indicative patterns, rather than using all of them we go over the list of indicative patterns, and keep a pattern only if it differs by at least 50% of its tokens from all already kept patterns.

Collecting pattern candidates: We start by simply searching a corpus for \(L\) patterns in which both terms from the input tuple \((a : a^*)\) appear in\(^1\). These are our pattern candidates that will be transferred to the filtering stage.

Filtering non-relationship-indicative patterns: we create a masked pattern from each pattern by replacing the appearance of the term \(a^*\) in the patterns with a mask. Then, we filter out patterns where the MLM cannot predict that \(a^*\) is the missing term. If the MLM cannot predict that \(a^*\) is missing, although \(a\) is in fact in the masked pattern, we have no reason to believe the pattern is indicative of the relationship between \(a\) and \(a^*\). Formally, We mark \(REP(p_i, t, t')\) as the pattern \(p_i\).

\(^1\)For simplicity reasons, we pick patterns where both \(a\) and \(a^*\) appear in exactly once.
where the term \( t \) is replaced with the term \( t' \) (which can also be a mask, marked as \([MASK]\)). We first derive a masked pattern from the pattern candidates by masking the term \( a^* \) out of the pattern. Then, we keep only the patterns where the MLM can restore \( a^* \), predicting it is the term most probable to fill the mask. We will define a score \( s(p_i, t) \) for a pattern \( p_i \) and a term \( t \) (recall \( R_{MLM}(t, m) \) is the rank (index) of the term \( t \) in the word completion list the MLM predicts for \( m \)):

\[
s(p_i, t) = R_{MLM}(t, REP(p_i, t, [MASK]))
\]  

(5.1)

For each pattern candidate \( p_i \) we compute \( s(p_i, a^*) \) and keep only the patterns for which \( s(p_i, a^*) = 1 \), meaning the MLM predicted \( a^* \) as the most probable mask replacement for the masked pattern acquired from \( p_i \). We will mark this group of chosen patterns as \( \text{relationship\_indicative\_patterns} \).

**Transforming the chosen patterns to make them indicative of \( b^* \):** we want to make the chosen patterns masked and indicative of the relationship \( r^* \), so we first replace each appearance of \( a^* \) with a mask:

\[
\text{masked\_patterns} = \{ REP(p_i, a^*, [MASK]) | p_i \in \text{relationship\_indicative\_patterns} \}
\]  

(5.2)

We want the indicative masked-patterns to be indicative of the relationship, but with respect to the question term \( b \), so we replace the appearances of the term \( a \) in every pattern with the term \( b \):

\[
\text{\( b^* \_indicative\_patterns \)} = \{ REP(m_i, a, b) | m_i \in \text{masked\_patterns} \}
\]  

(5.3)

For example, if \((a : a^*) = ("Paris", "France") \) and \( b = "Berlin" \), we will first query the corpus for patterns containing the terms "Paris" and "France". One such pattern may be "The capital of France is Paris". Then, we will compute \( s(p_i, a^*) \) by masking out the term "France" from the acquired patterns, obtaining masked-patterns like "The capital of ___ is Paris" and searching for the index of \( a^* \) in the MLM’s completion suggestions\(^2\). We will filter out patterns \( p_i \) where the MLM can’t predict that \( a^* \) is the missing term, meaning that \( s(p_i, a^*) \neq 1 \). If the above pattern is not filtered and therefore chosen to be indicative, we will finally mask out the term "France" and replace the term "Paris" with the term "Berlin" to form the indicative pattern "The capital of ___ is Berlin".

### 5.2.2 Enhanced pattern picking method

While using only the filter suggested in section 5.2.1 for filtering out non-relationship indicative patterns some of the patterns we acquire will in fact indicate the relationship

\(^2\)Notice that this method requires the term \( a^* \) to be in the MLM’s vocabulary, otherwise it will never be found as an MLM completion for any pattern.
between $a$ and $a^*$, not all of them will. We might also acquire patterns where $a^*$ is likely to appear in, without respect to $a$. For example, if $(a,a^*) = ("Paris","France")$ the pattern “France has borders with Italy Spain and Germany, and Paris is considered the fashion capital of the world.” will probably not be filtered out because it will lead to the masked pattern “___has borders with Italy Spain and Germany, and Paris is considered the fashion capital of the world.”. The term *France* is probable to be most likely to fill this pattern when querying the MLM (because France is the only country that shares borders with Italy Spain and Germany) and the pattern will probably be kept. This is problematic because the pattern does not catch the relationship between *Paris* and *France*, it just has a strong connection to the term *France* and therefore the MLM is able to restore it\textsuperscript{3}. If this pattern is indeed chosen as indicative, when we will mask out “France” and switch “Paris” with a different capital city (e.g. “Berlin”), the masked pattern “___has borders with Italy Spain and Germany, and Berlin is considered the fashion capital of the world.” will be formed, which is definitely NOT indicative of “Germany” like we want it to be. In order to address this problem, we will add another filter to the candidate patterns filtering process, apart from just filtering out patterns where $s(p_i,a^*) \neq 1$.

Intuitively, we will filter out patterns if $a^*$ can be restored by the MLM without the MLM seeing $a$ in the pattern. If it can, this means the pattern is indicative of $a^*$ without respect to $a$ and therefore if we were to keep this pattern, when we would replace $a$ with $b$ later on, it will still be indicative of $a^*$ and not of $b^*$ as we want it to be. We will apply this filter by first masking out BOTH $a$ and $a^*$ from the pattern, and then checking if the MLM can restore the term $a^*$ (meaning, it appears first on the MLM replacements list of the masked place it appeared in originally). If it can, $a^*$ is linked to the pattern not only because of the relationship with $a$ but due to some other features of the pattern.

Let $m$ be a masked pattern with two masked places. Let $MLM_{first}(m)$ be the word completions (mask replacements) predicted by the MLM for the first masked place of the masked pattern $m$ ranked by their probability, and $MLM_{second}(m)$ be the word completions (mask replacements) predicted by the MLM for the second masked place of the masked pattern $m$ ranked by their probability. Let $R_{MLM_{first}}(t,m)$ be the rank (index) of the term $t$ in $MLM_{first}(m)$. We define a double masked pattern from a pattern and two terms by simply replacing both terms with a mask:

$$double\_\text{masked}(p_i,t,t') = \text{REP}(\text{REP}(p_i,t',[\text{MASK}]),t,[\text{MASK}]) \quad (5.4)$$

\textsuperscript{3}This specific pattern is an extreme example, but we observed during our work many cases where there is in fact a strong connection between the pattern and the term $a^*$ even when disregarding the term $a$. 

24
Figure 5.1: The analogy solving pattern filtering process.

We demonstrate the process for the two-tuple \((a, a^*) = ("paris", "france")\). In the first box, collected patterns that contain both \(a\) and \(a^*\) are shown. In the second and third boxes, the first filter is applied by masking out \(a^*\) and keeping only patterns where the MLM can restore it. For the base method (described in section 5.2.1), the patterns in box 3 will be the ones kept and turned into indicative of \(b^*\). For the enhanced method (described in section 5.2.2) a second filter is applied (box’s 4 and 5) by masking out both \(a\) and \(a^*\) and keeping a pattern only if the MLM cannot restore \(a^*\) in this case. After choosing the patterns we will use, we transform them into masked, \(b^*\) indicative patterns in box number 6.

We define a new pattern score \(s'\) with respect to a pattern and two terms:

\[
s'(p_i, t, t') = R_{LM_{first}}(t', \text{double\_masked}(p_i, t, t'))
\]  

(5.5)

Finally, for each pattern candidate \(p_i\), assuming w.l.o.g.\(^4\) that \(a\) appears before \(a^*\) in \(p_i\), we compute \(s'(p_i, a, a^*)\). We filter out a pattern \(p_i\) if \(s'(p_i, a, a^*) = 1\).

To sum things up, to acquire the enhanced indicative patterns we first take patterns containing both \(a\) and \(a^*\) from the corpus as candidate patterns. We filter out patterns where the MLM can’t restore \(a^*\) by deleting patterns where \(s(p_i, a^*) \neq 1\). Then we apply a second filter on the remaining candidate patterns by filtering out patterns where \(a^*\) can be restored without the MLM seeing \(a\) in the pattern. We dis-consider patterns if \(s'(p_i, a, a^*) = 1\). Finally, we take the chosen patterns, mask them, and make them indicative of \(b^*\) as described in section 5.2.1. This procedure is demonstrated in fig. 5.1. Examples for indicative masked patterns found using this method can be found at appendix B.2.

\(^4\)if \(a\) appears after \(a^*\) we will use \(R_{LM_{second}}\) instead of \(R_{LM_{first}}\) in the \(s'\) definition.
### Table 5.2: Analogy Dataset statistics

<table>
<thead>
<tr>
<th>Category</th>
<th># in original dataset</th>
<th># after filtering*</th>
<th>reported**</th>
</tr>
</thead>
<tbody>
<tr>
<td>capital:common_country</td>
<td>506</td>
<td>506</td>
<td>yes</td>
</tr>
<tr>
<td>capital: country</td>
<td>4524</td>
<td>2771</td>
<td>yes</td>
</tr>
<tr>
<td>country:currency</td>
<td>866</td>
<td>30</td>
<td>no</td>
</tr>
<tr>
<td>city:state</td>
<td>2467</td>
<td>2467</td>
<td>yes</td>
</tr>
<tr>
<td>male:female</td>
<td>506</td>
<td>380</td>
<td>yes</td>
</tr>
<tr>
<td>total</td>
<td>8869</td>
<td>6154</td>
<td>NA</td>
</tr>
</tbody>
</table>

*Filtering out four-tuples \((a : a^*, b : b^*)\) where \(a^*\) or \(b^*\) are not in Bert’s vocabulary and where \(a, a^*, b, b^*\) are not in the top-60,000 most common in the w2v vocabulary. **The country:currency category is very small after applying the filters, so it is not used in later experiments.

5.2.3 pattern generalization via MLM query

Having identified indicative patterns, we now turn to fill them: finding a single term that fits them the best. Similarly to section 4.2.2, we will fill the indicative pattern found by simply querying an MLM\(^5\). Each indicative pattern \(m_i\) naturally provides a ranked list of candidate terms \(LM(m_i) = t_1, ..., t_{|V|}\), where \(V\) is the MLM’s vocabulary and each term \(t_j\) is scored by its pattern-conditional probability. We combine the term scores from all chosen indicative patterns using a product of experts approach, scoring each term by the product of probabilities (sum of log probabilities) assigned to it by each context. Let \(p_{MLM}(t|m_i)\) be the probability assigned to vocabulary term \(t\) in pattern \(m_i\). The term score is:

\[
\text{score}(t) = \sum_{m_i \in \text{indicative patterns}} \log p_{MLM}(t|m_i) \tag{5.6}
\]

As opposed to the TSE task, for the analogy solving task we want to find a single term that fits the patterns. Therefore, our analogy prediction will be the term with the highest \(\text{score}(t)\).

5.3 Experiments and Results

**Setup.** In our experiments we use BERT [DCLT19] as the MLM, and English Wikipedia as the corpus. We measure performance using a simple precision measure. Based on preliminary experiments, we use \(L = 2000\) when reporting the main results.\(^6\)

**Dataset.** We use the google analogy dataset\(^7\) containing analogies from several different categories. The dataset contains both semantic categories (e.g. country-to-capital) and grammatical categories (e.g adjective-to-adverb). For our purpose, we use only the semantic categories. For a fair comparison, we use only four-tuples

---

\(^5\)This means our predicted term, \(b^*\), will be a term from the MLM’s vocabulary.

\(^6\)\(L\) is the number of candidate patterns collected from the corpus. See Additional experiments for a justification of this parameter choice.

\(^7\)https://github.com/nicholas-leonard/word2vec/blob/master/questions-words.txt
(a : a∗, b : b∗) that all compared methods have a chance to solve, dictating a∗ and b∗ are a part of Bert’s vocabulary and all four terms (a, a∗, b and b∗) are in the w2v vocabulary (see Compared methods). The original dataset contains 5 semantic categories (capital:common_country, capital:country, country:currency, city:state and male:female family members) but the country:currency category shrinks down to only 30 4-tuples after applying the filters, so we do not use it for later experiments. Dataset statistics can be found at fig. 5.2.

Compared methods. We compare our base method (where indicative patterns are acquired as described in section 5.2.1), reported as Base, and our enhanced pattern picking method (Enhanced) described in section 5.2.2 to a distribution w2v-analogy extractor (reported as W2V) like the one described in [MCCD13]. For this method, we use vectors pre-trained on a large corpus and solve the analogies like the classic distribution based analogy solution as described in section 5.1. For the word vectors, we use a vocabulary size of 60,000. We do so because using a larger vocabulary makes solving the analogy harder, so we want the vocabulary size of the compared methods to be similar. We also report a fourth method, Naive_Enhanced, which uses our base method as described in section 5.2.1 to choose indicative patterns, but in order to solve the problem described in section 5.2.2, it disallows the predicted b∗ to be a∗ by using the following heuristic: if the term returned by the “filling the patterns” method (section 5.2.3) is a∗, we simply take the next ranked term. We report this method to show that our enhanced method doesn’t only avoid mistakes from the type of choosing a∗, it also picks better quality indicative patterns.

Main results are reported in fig. 5.3. Our Enhanced method achieves the best accuracy on 3 out of 4 categories. The total accuracy on all 4-tuples in the dataset is by far the best with our enhanced method. Another result is that the enhanced method works better than the naive enhanced one, meaning the filter added for the enhanced method does not only eliminate mistakes such as choosing a∗ as the analogy solution but also helps us keep the better, more relationship indicative patterns. The only category
Table 5.4: Tuning the number of sentences parameter for the Base method
Results are reported as (accuracy, average number of indicative patterns chosen after filtering the patterns).

that our method does not perform best on is the smallest category, male:female family members. Error analysis we conduct on this category leads us to a limitation of our method: in order for our method to perform well, the relationship of the analogy terms needs to be 'local': one that is caught in text patterns containing both analogy terms (e.g. capital_of or city_in_state), as opposed to relationships that are dictated by the analogy terms' local relationships with other entities or objects, and not directly with one another (male:female family members). This explains the relatively poor performance of our method on this category. Examples for analogy solutions generated using our method can be found at appendix B.2.

Additional experiments. In fig. 5.4 and fig. 5.5 we answer the question: how many sentences that contain a and a∗ do our methods need in order to perform well (assuming a pre-trained MLM)? fig. 5.4 reports on Base and fig. 5.5 reports on Enhanced. It can be seen that while as expected, the more sentences we use the better results we get, as few as 100 sentences can lead to useful results. 500 sentences lead to near-optimal results while using 1000 or 2000 sentences barely changes the results for the enhanced method. We also report in these tables the number of sentences actually used as indicative after the pattern filters are applied. An interesting observation is that around 40% of the patterns that survive the first filter (and are used for the Base method) are dropped when applying the second filter for the enhanced method, meaning they are indicative of a∗ regardless of a appearing in them.

Table 5.5: Tuning the number of sentences parameter for the Enhanced method
Results are reported as (accuracy, average number of indicative patterns chosen after filtering the patterns).
5.4 Conclusions

We introduce an MLM-based analogy solving method, reaching state-of-the-art results. The method uses the power of MLM predictions to generate patterns indicative of the concept term (indicated by the known analogy terms), and then to find the concept term. Beyond strong results on the analogy solving task, our method is also transparent: the relationship-indicative patterns that were used to find the concept term can be investigated and understood. We demonstrate a novel use of pre-trained MLMs, using their predictions directly rather than relying on their states for fine-tuning.
Chapter 6

Final Remarks

We have researched and investigated methods for semantic extraction tasks, combining two common approaches that, as far as we know, have not been combined yet. Our main contributions are as follows:

- A general workflow for semantic extraction tasks (chapter 3).
- A method for Term Set Expansion (chapter 4).
- A masked pattern similarity measure (section 4.2.3).
- A Term Set Expansion data-set (section 4.3).
- A method for Analogy Solving (chapter 5).

Our work implies some future research questions. First and foremost, what other semantic extraction tasks can our workflow be used for? Named Entity Recognition (NER) and Relation Extraction (RE) are some of the candidate tasks. Second, can we generate indicative masked patterns from the model instead of just searching for pattern candidates in a corpus? If we can generate masked patterns, in a way that will maximize the probability to find terms we desire as mask replacements, we could acquire better indicative patterns, and thus boost performance. Also, it will lead to better treatment for rare words and terms for which a sufficient amount of patterns containing them might not be available.

A field where progress is incessantly made, which our work can benefit from directly is Language Modeling and Masked Language Modeling. Future work in this field, combined with the continuing surge in computational power, is expected to produce better, more knowledgeable LMs which our methods, among others, will benefit from.
Appendix A

Additions to Chapter 4

A.1 Finding candidate terms for TSE

For our first method, MPB1, the candidate terms we score are just the terms in the MLM’s vocabulary. For our second method, MPB2, we want to score candidates which are not in this vocabulary as well. Hence, we need a way to acquire these candidates. Scoring all possible terms is prohibitive, so we seek an efficient method to acquire a high-recall group of candidates for the desired semantic class. We get this using a simple distributional set-expander: we compute the mean vector for words in our seed set and look for the top-k neighbors in a distributional space.

Specifically, we use sense2vec pre-trained vectors. Sense2vec [TML15] is a misleadingly-named algorithm from the w2v-family [MCCD13] that models each term as “term/part of speech”. This allows it, for example, to learn different representations for “duck/verb” and “duck/noun”.

More importantly, the pre-trained sense2vec vectors distributed by explosion.ai\(^1\) are trained over a large and diverse English corpus (Reddit posts and comments from 2015 and 2019), and its vocabulary includes not only single words but also multi-word units (NP-chunks and named entities).

A.2 TSE dataset example

In this section, we give an example class from the TSE dataset we generated. The data is not case sensitive. We report on the NFL team semantic class. The full dataset is available at https://github.com/guykush/TermSetExpansion-MPB/.

NFL teams set

{Bills, the_Bills, Buffalo_Bills, the_Buffalo_Bills}
{Dolphins, the_Dolphins, Miami_Dolphins, the_Miami_Dolphins, Phins, the_phins}

\(^1\)https://explosion.ai/demos/sense2vec
{Patriots, the_Patriots, New_England_Patriots, the_New_England_Patriots, Pats, the_pats}
{Jets, the_Jets, New_York_Jets, NY_Jets, the_New_York_Jets, the_NY_Jets}
{Ravens, the_Ravens, Baltimore_Ravens, the_Baltimore_Ravens}
{Bengals, the_Bengals, Cincinnati_Bengals, the_Cincinnati_Bengals}
{Browns, the_Browns, Cleveland_Browns, the_Cleveland_Browns}
{Steelers, the_Steelers, Pittsburgh_Steelers, the_Pittsburgh_Steelers}
{Texans, the_Texans, Houston_Texans, the_Houston_Texans}
{Colts, the_Colts, Indianapolis_Colts, the_Indianapolis_Colts}
{Jaguars, the_Jaguars, Jacksonville_Jaguars, the_Jacksonville_Jaguars, Jags, the_jags}
{Titans, the_Titans, Tennessee_Titans, the_Tennessee_Titans}
{Broncos, the_Broncos, Denver_Broncos, the_Denver_Broncos}
{Chiefs, the_Chiefs, Kansas_City_Chiefs, kc_chiefs, the_Kansas_City_Chiefs, the_kc_chiefs}
{Chargers, the_Chargers, Los_Angeles_Chargers, LA_chargers, the_Los_Angeles_Chargers, the_LA_chargers}
{Raiders, the_Raiders, Oakland_Raiders, the_Oakland_Raiders}
{Cowboys, the_Cowboys, Dallas_Cowboys, the_Dallas_Cowboys}
{Giants, the_Giants, New_York_Giants, NY_Giants, the_New_York_Giants, the_NY_Giants}
{Eagles, the_Eagles, Philadelphia_Eagles, the_Philadelphia_Eagles}
{Redskins, the_Redskins, Washington_Redskins, the_Washington_Redskins}
{Bears, the_Bears, Chicago_Bears, the_Chicago_Bears}
{Lions, the_Lions, Detroit_Lions, the_Detroit_Lions}
{Packers, the_Packers, Green_Bay_Packers, the_Green_Bay_Packers}
{Vikings, the_Vikings, Minnesota_Vikings, the_Minnesota_Vikings, Vikes, the_vikes}
{Falcons, the_Falcons, Atlanta_Falcons, the_Atlanta_Falcons, Falcs, the_falcs}
{Panthers, the_Panthers, Carolina_Panthers, the_Carolina_Panthers}
{Saints, the_Saints, New_Orleans_Saints, the_New_Orleans_Saints}
{Buccaneers, the_Buccaneers, Tampa_Bay_Buccaneers, the_Tampa_Bay_Buccaneers, Bucs, the_bucs}
{Cardinals, the_Cardinals, Arizona_Cardinals, the_Arizona_Cardinals, cards, the_cards}
{Rams, the_Rams, Los_Angeles_Rams, LA_rams, the_Los_Angeles_Rams, the_LA_rams}
{49ers, the_49ers, San_Francisco_49ers, Niners, 9ers, 49rs, the_Niners, the_9ers, the_49rs}
{Seahawks, the_Seahawks, Seattle_Seahawks, the_Seattle_Seahawks}
A.3 TSE indicative pattern examples

In this section, we give examples of indicative patterns our method finds for different seed-terms. We also report the $maxRank(m_i)$ of the masked pattern, which induced the choosing of it as indicative as described in (section 4.2.1).

Seed: \{“bush”, “obama”, “adams”\}, class: US presidents

- $m_1 = \text{``and andrew johnson are the only former presidents to serve in congress.''}, maxRank(m_1) = 9$
- $m_2 = \text{``is inaugurated the 43rd president of the united states of america.''}, maxRank(m_2) = 15$
- $m_3 = \text{``felt that his own election as president would vindicate his father.''}, maxRank(m_3) = 17$
- $m_4 = \text{``1988 ____began planning for a presidential run after the 1984 election.''}, maxRank(m_4) = 17$
- $m_5 = \text{``president ____meeting in the oval office with secretary gates.''}, maxRank(m_5) = 22$

Seed: \{“vikings”, “colts”, “dolphins”\}, class: NFL teams

- $m_1 = \text{``punter david lee booted a 51-yard kick that pinned the jets back at their own 4-yard line.''}, maxRank(m_1) = 6$
- $m_2 = \text{``the ____advanced to the super bowl by beating the cincinnati bengals and the oakland raiders in the playoffs at memorial stadium''}, maxRank(m_2) = 6$
- $m_3 = \text{``and devin hester did so for the chicago bears against the indianapolis ____in super bowl xli''}, maxRank(m_3) = 7$
- $m_4 = \text{``and was sacked on third down by ____linebacker dennis gaubatz for a 2-yard loss.''}, maxRank(m_4) = 7$
- $m_5 = \text{``the 2006 indianapolis ____honored at the white house for their super bowl victory.''}, maxRank(m_5) = 7$

Seed: \{“cubs”, “mets”, “phillies”\}, class: MLB teams

- $m_1 = \text{`` ____rookie third baseman kris bryant was the first to hit the new left field videoscreen with his home run.''}, maxRank(m_1) = 3$
• \( m_2 = \text{"making no significant contribution to the 1973 pennant-winning campaign; he was sold to the texas rangers mid-season."}, \)
  \( \text{maxRank}(m_2) = 3 \)

• \( m_3 = \text{"when a ___ player hits a home run a recording of kalas' famous 'that ball is outta here!' home run call is played."}, \)
  \( \text{maxRank}(m_3) = 3 \)

• \( m_4 = \text{"behind a dramatic home run by ___catcher mike piazza."}, \)
  \( \text{maxRank}(m_4) = 4 \)

• \( m_5 = \text{"including his status as the ___ all-time home run leader."}, \)
  \( \text{maxRank}(m_5) = 4 \)

Seed: \{“baghdad”, “tbilisi”, “berlin”\}, class: Capitals

• \( m_1 = \text{"azerbaijan has an embassy in ___."}, \)
  \( \text{maxRank}(m_1) = 29 \)

• \( m_2 = \text{"was hit by an sa-14 'gremlin' missile after takeoff from ____international airport."}, \)
  \( \text{maxRank}(m_2) = 34 \)

• \( m_3 = \text{"dominican republic is accredited to turkey from its embassy in ___."}, \)
  \( \text{maxRank}(m_3) = 42 \)

• \( m_4 = \text{"but they reported to his central government in ___."}, \)
  \( \text{maxRank}(m_4) = 47 \)

• \( m_5 = \text{"but in april 2005 a new embassy opened in ___."}, \)
  \( \text{maxRank}(m_5) = 53 \)

Seed: \{“japan”, “jordan”, “kenya”\}, class: Countries

• \( m_1 = \text{"at the mayflower hotel (1957) eisenhower applied the doctrine in 1957–58 by dispensing economic aid to shore up the kingdom of ___."}, \)
  \( \text{maxRank}(m_1) = 19 \)

• \( m_2 = \text{"among these include: united nations development programme representative of ___."}, \)
  \( \text{maxRank}(m_2) = 25 \)

• \( m_3 = \text{"azerbaijan • bahrain • cyprus (including disputed northern cyprus)
  • georgia • iraq • israel • ____ • kuwait • lebanon • oman • palestinian territories qatar • saudi arabia • syria • turkey • united arab emirates
  • yemen : ‘‘'caucasus''' (a region considered to be in both asia and europe.''}, \)
  \( \text{maxRank}(m_3) = 26 \)

• \( m_4 = \text{"ethiopia appointed its first ambassador to ___."}, \)
  \( \text{maxRank}(m_4) = 27 \)
• $m_5 = \text{``according to \textquotesingle\textquotesingle's minister for water.\textquotesingle\textquotesingle}, \text{maxRank}(m_5) = 29$

Seed: \{\text{``moldova'', \text{``romania'', \text{``spain''}}\}, \text{class: Europien Countries}

• $m_1 = \text{``in a european championship qualifier against ____\textquotesingle\textquotesingle}, \text{maxRank}(m_1) = 7$

• $m_2 = \text{``106 - the south-western part of dacia (modern ____\textquotesingle\textquotesingle} becomes a roman province: roman dacia\textquotesingle\textquotesingle}, \text{maxRank}(m_2) = 11$

• $m_3 = \text{``they can be found in the coat of arms of both romania and ____\textquotesingle\textquotesingle}, \text{maxRank}(m_3) = 13$

• $m_4 = \text{``following the accession of ____and bulgaria in january 2007\textquotesingle\textquotesingle}, \text{maxRank}(m_4) = 15$

• $m_5 = \text{``among the visigoths settled in lower moesia (now part of bulgaria and ____\textquotesingle\textquotesingle)}\textquotesingle\textquotesingle, \text{maxRank}(m_5) = 18$

Seed: \{\text{``rap'', \text{``pop'', \text{``techno''}}\}, \text{class: Music genres}

• $m_1 = \text{``diplo fused house music with ____and dance/pop\textquotesingle\textquotesingle}, \text{maxRank}(m_1) = 7$

• $m_2 = \text{``often including djs and ____-style vocals\textquotesingle\textquotesingle}, \text{maxRank}(m_2) = 8$

• $m_3 = \text{``hip-hop also made its mark on country music with the emergence of country ____\textquotesingle\textquotesingle}, \text{maxRank}(m_3) = 9$

• $m_4 = \text{``radio eska (____and dance music) \textquotesingle\textquotesingle}, \text{maxRank}(m_4) = 9$

• $m_5 = \text{``thrash has been described as a form of \text{``urban blight music''} and \text{``a palefaced cousin of ____\textquotesingle\textquotesingle}\textquotesingle\textquotesingle, \text{maxRank}(m_5) = 9$

Seed: \{\text{``oregon'', \text{``alaska'', \text{``alabama''}}\}, \text{class: US states}

• $m_1 = \text{``members of the nine seats on the supreme court of ____and all ten seats on the state appellate courts are elected to office\textquotesingle\textquotesingle}, \text{maxRank}(m_1) = 5$

• $m_2 = \text{``it encompasses the us states of ____\textquotesingle\textquotesingle}, \text{maxRank}(m_2) = 9$

• $m_3 = \text{``the rv was stopped by an ____state trooper\textquotesingle\textquotesingle}, \text{maxRank}(m_3) = 9$

• $m_4 = \text{``until ***mask*** became a state in 1959\textquotesingle\textquotesingle}, \text{maxRank}(m_4) = 12$

• $m_5 = \text{`` ***cost of living*** the cost of goods in ____has long been higher than in the contiguous 48 states\textquotesingle\textquotesingle}, \text{maxRank}(m_5) = 14$
A.4  Term Set Expansion Examples

The following are term set expansion examples produced by our methods. Parameters used for this expansions are the ones reported in section 4.3 as the main results parameters. Specifically, Bert is used as the MLM, and Wikipedia is used as the corpus. For MPB1 we use $\ell = 160$ and $L = 2000/k$ and for MPB2 we use $\ell = 20$ and $L = 2000/k$. For each expansion, we report the top 100 terms in the returned list, the seed terms used, and whether MPB1 or MPB2 was used.

Seed: \{“bush”, “obama”, “adams”\}, MPB1


Seed: \{“bush”, “obama”, “adams”\}, MPB2

Seed: \{“vikings”, “colts”, “dolphins”\}, MPB1


Seed: \{“vikings”, “colts”, “dolphins”\}, MPB2

baker mayfield

Seed: \{"cubs", "mets", "phillies"\}, MPB1


Seed: \{"cubs", "mets", "phillies"\}, MPB2


Seed: \{"baghdad", "tbilisi", "berlin"\}, MPB2

37. kuwait city 38. beirut 39. niamey 40. pristina 41. asunción 42. nicosia 43. izmir 44. istanbul 45. islamabad 46. athens 47. port louis 48. rome 49. tegucigalpa 50. tirana 51. warsaw 52. prague 53. tunis 54. casablanca 55. kyiv 56. cetinje 57. riyadh 58. belgrade 59. haifa 60. antananarivo 61. munich 62. mexico city 63. ottawa 64. thimphu 65. skopje 66. beijing 67. abu dhabi 68. sarajevo 69. reykjavik 70. quito 71. hungary 72. dhaka 73. stockholm 74. mosul 75. havana 76. bratislava 77. tokyo 78. phnom penh 79. karachi 80. addis ababa 81. manama 82. odessa 83. saxony 84. ljubljana 85. tallinn 86. buenos aires 87. yaoundé 88. conakry 89. tripoli 90. khartoum 91. st petersburg 92. helsinki 93. strasbourg 94. zagreb 95. kigali 96. port-au-prince 97. kinshasa 98. jakarta 99. bogotá 100. gaborone

Seed: {“japan”, “jordan”, “kenya”}, MPB2

1. kenya 2. lesotho 3. qatar 4. benin 5. botswana 6. bahrain 7. burundi 8. south korea 9. zimbabwe 10. south sudan 11. malayasia 12. guyana 13. cameroon 14. bhutan 15. algeria 16. taiwan 17. greece 18. namibia 19. tanzania 20. somalia 21. eswatini 22. eritrea 23. burma 24. the united arab emirates 25. dr congo 26. armenia 27. fiji 28. pakistan 29. kazakhstan 30. ivory coast 31. north korea 32. indonesia 33. uganda 34. mauritania 35. papua new guinea 36. angola 37. hungary 38. guinea-bissau 39. thailand 40. zambia 41. djibouti 42. ghana 43. singapore 44. india 45. libya 46. equatorial guinea 47. burkina faso 48. chad 49. ukraine 50. china 51. japan 52. belarus 53. vanuatu 54. antiqua and barbuda 55. uzbekistan 56. afghanistan 57. kuwait 58. senegal 59. bangladesh 60. yemen 61. uruguay 62. latvia 63. myanmar 64. oman 65. the far east 66. brunei 67. many other countries 68. rwanda 69. azerbaijan 70. the dominican republic 71. turkey 72. belize 73. alaska 74. costa rica 75. barbados 76. iran 77. the czech republic 78. tonga 79. malawi 80. sierra leone 81. saudi arabia 82. foreign countries 83. lebanon 84. germany 85. several countries 86. south asia 87. laos 88. russia 89. dominica 90. seychelles 91. ethiopia 92. latin america 93. slovenia 94. suriname 95. albania 96. trinidad and tobago 97. liberia 98. the united states 99. mozambique 100. bosnia and herzegovina

Seed: {“moldova”, “romania”, “spain”}, MPB2

Seed: {“rap”, “pop”, “techno”}, MPB2


Seed: {“oregon”, “alaska”, “alabama”}, MPB2

surrounding states 72. other states 73. tulsa 74. nys 75. upstate ny 76. washington
77. northern california 78. floridian 79. louisville 80. ocala 81. albuquerque 82. salt
lake city 83. auburn 84. new orleans 85. spokane 86. asheville 87. jacksonville 88.
topeka 89. ohio state 90. madison 91. colorado springs 92. wichita 93. the deep south
94. huntsville 95. savannah 96. central florida 97. southern states 98. gainesville 99.
buffalo 100. columbia
Appendix B

Additions to Chapter 5

B.1 Analogy solving indicative pattern examples

In this section, we give examples of indicative patterns our method finds for different semantic relationships. We report patterns found with the enhanced method as described in section 5.2.2. We report the input analogy terms $a$ and $a^*$ used to search for the relationship indicative pattern. We also report the question term $b$ used to assemble the masked indicative pattern and $b^*$, the desired concept term which the patterns are supposed to be indicative of. Notice similar patterns can be formed such that they will be indicative for different $b^*$ terms as described in appendix B.2 by simply replacing the term $b$ with a different term that shares the concept relationship with the new $b^*$.

$$(a, a^*) = (\text{'athens', 'greece'}), (b, b^*) = (\text{'madrid', 'spain'})$$

concept relationship: $a$ is the capital of $a^*$

- $m_1 = \text{``the embassy of indonesia in madrid was opened in 1994 and was followed by the establishment of the embassy of ____ in jakarta in 1997.''}$
- $m_2 = \text{``algeria is represented in ____ by its embassy in madrid.''}$
- $m_3 = \text{``with the biggest airport of ____the madrid international airport.''}$
- $m_4 = \text{``western ____ and the capital madrid.''}$
- $m_5 = \text{``the municipality (city) of madrid is the most populous in ____.''}$

$$(a, a^*) = (\text{'abuja', 'nigeria'}), (b, b^*) = (\text{'ankara', 'turkey'})$$

concept relationship: $a$ is the capital of $a^*$

- $m_1 = \text{``the capital of ____ is ankara.''}$
• $m_2 = "\text{airways had its headquarters at airways house in ankara at the time of dissolution."}^

• $m_3 = "\text{ankara chapter of the chemical society of where he hosted several national and international chemistry professional events."}^

• $m_4 = "\text{ireland is represented in through its embassy in ankara."}^

• $m_5 = "\text{the 4th edition of ankara bantaba honored 100 personalities who are key players to tourism development in }"^$

$(a, a^*) = \text{chicago', 'illinois', (b, b^*) = ('philadelphia', 'pennsylvania')}$

\text{concept relationship: } a \text{ is a city in the state } a^*$

• $m_1 = "\text{and the university of at philadelphia are classified as }"^$

• $m_2 = "\text{more than half the population of the state of lives in the philadelphia metropolitan area."}^

• $m_3 = "\text{the near west side holds the university of at philadelphia and was once home to oprah winfrey's harpo studios."}^

• $m_4 = "\text{geography= downtown and the north side with beaches lining the waterfront philadelphia is located in northeastern on the southwestern shores of freshwater lake michigan."}^

• $m_5 = "\text{walter payton college prep high school is ranked number one in the city of philadelphia and the state of }"^$

$(a, a^*) = \text{boy', 'girl', (b, b^*) = ('brother', 'sister')}$

\text{concept relationship: } a \text{ is the male version of } a^*$

• $m_1 = "\text{1 cute faces blonde and orange-haired brother."}^

• $m_2 = "\text{geek brother with numbers and chubby with food dishes."}^

• $m_3 = "\text{young hip-hop brother and competing each other until the end."}^

• $m_4 = "\text{amy put a blue ribbon on the brother and a pink on the }"^$

• $m_5 = "\text{the sex of the baby is indicated by the color of the beads-blue for a brother and pink for a }"^$
B.2 Solved analogies

The following are analogy solution examples. For these results we use $L = 2000$ as in the main results reported at section 5.3. We use the enhanced method, Bert is used as the MLM, and Wikipedia is used as the corpus. For each analogy the report has the following structure: correct/incorrect: $a, a^*, b, b^*$ returned by our method. if the analogy solution is incorrect we also add the wanted term (the oracle $b^*$). We report 100 analogies for each category. Example reports: correct: athens greece baghdad iraq For this report $(a, a^*) = \text{'athens', 'greece'},$ and $(b, b^*) = \text{('baghdad', 'iraq')}. The b* the method returned is the correct one. incorrect: berlin germany london london wanted: england For this report $(a, a^*) = \text{'berlin', 'germany'},$ and $(b, b^*) = \text{('london', 'london')}. The b* the method returned is incorrect, the correct answer would be 'england'.

Concept relationship: $a$ is the capital of $a^*$, common counties


**Concept relationship: a is the capital of a**, non-common counties

many, correct: accra ghana bern switzerland, correct: accra ghana bishkek kyrgyzstan,
correct: accra ghana bratislava slovakia, correct: accra ghana brussels belgium, correct: 
accra ghana bucharest romania, correct: accra ghana budapest hungary, correct: accra 
ghana cairo egypt, correct: accra ghana canberra australia, correct: accra ghana cara-
cas venezuela, correct: accra ghana copenhagen denmark, correct: accra ghana dakar 
seNEGAL, correct: accra ghana damascus syria, correct: accra ghana dhaka bangladesh, 
correct: accra ghana doha qatar, correct: accra ghana dublin ireland, correct: algiers 
algeria amman jordan, correct: algiers algorea ankara turkey, correct: algiers algorea as-
tana kazakhstan, correct: algiers algorea athens greece, correct: algiers algorea baghdad 
iraq, correct: algiers algorea baku azerbaijan, correct: algiers algorea bangkok thailand, 
correct: algiers algorea beijing china, correct: algiers algorea beirut lebanon, correct: 
algiers algorea belgrade serbia, correct: algiers algorea berlin germany, correct: algiers 
algeria bern switzerland, correct: algiers algorea bishkek kyrgyzstan, correct: algiers 
algeria bratislava slovakia, correct: algiers algorea brussels belgium, correct: algiers 
algeria bucharest romania, correct: algiers algorea budapest hungary, correct: algiers 
algeria cairo egypt, correct: algiers algorea canberra australia, correct: algiers algorea 
caracas venezuela, correct: algiers algorea copenhagen denmark, correct: algiers algorea 
dakar senegal, correct: algiers algorea damascus syria, correct: algiers algorea dhaka 
bangladesh, correct: algiers algorea doha qatar, correct: algiers algorea dublin ireland, 
correct: algiers algorea gaborone botswana, correct: amman jordan ankara turkey, cor-
rect: amman jordan astana kazakhstan, correct: amman jordan athens greece, correct: 
amman jordan baghdad iraq, correct: amman jordan baku azerbaijan, correct: amman 
jordan bangkok thailand, correct: amman jordan beijing china, correct: amman jordan 
beirut lebanon, correct: amman jordan belgrade serbia, correct: amman jordan berlin 
germany, correct: amman jordan bern switzerland, correct: amman jordan bishkek kyr-
gyzstan, correct: amman jordan bratislava slovakia, correct: amman jordan brussels 
belgium, correct: amman jordan bucharest romania, correct: amman jordan budapest 
hungary, correct: amman jordan cairo egypt, correct: amman jordan canberra australia

Concept relationship: a is a city in the state a∗
correct: chicago illinois houston texas, correct: chicago illinois philadelphia pennsylva-
nia, correct: chicago illinois phoenix arizona, correct: chicago illinois dallas texas, cor-
rect: chicago illinois jacksonville florida, correct: chicago illinois indianapolis indiana, 
correct: chicago illinois austin texas, correct: chicago illinois detroit michigan, cor-
rect: chicago illinois memphis tennessee, correct: chicago illinois boston massachusetts, 
correct: chicago illinois seattle washington, correct: chicago illinois denver colorado, 
correct: chicago illinois baltimore maryland, correct: chicago illinois nashville tenn-
nessee, correct: chicago illinois louisville kentucky, correct: chicago illinois milwaukee 
wisconsin, correct: chicago illinois portland oregon, correct: chicago illinois tucon 
arizona, correct: chicago illinois fresno california, correct: chicago illinois sacramento
concept relationship: $a$ is the male version of $a^*$

Bibliography


המתאימים לתבניות אלה: מציאת מושגים שישינו יופי בהבנת תכלית.

בנוסף, לקטט בפואטיקה מביאות תבניות קלאסיות שבושלו במעה של א던 חומת תבניות. אגם ממוסדות את השימוט בחפיפה של התבניות וה-php הפשיבי של תבניות תופסות. האstdClassים הם

הרחבת קבוצת מושגים: ההינתן קבוצת כותנה של מונחים השניים לאיתת הקבוצת הסמינית. רצף הלוחרים אוזה לכלל המושגים-span

ולמשל, אם נבדל את הקבוצת המוסרת

אות החול או את הקבוצת כל החולות. המושמות הסמיניות השתייכי לעילה בנגדים

את השימוע בצפיפות השורות הצפופים הם פתרון אנלוגיות: בהינתן זוג מושגים שבושי קיים קשר

כלשה, למה ("ישראל", "ירושלים") פופולע שלושי, למה "ספרד", נרצה להפוךหวה ראשית, אלא בין שניים שניים והрош את הקבר לשון המושגים שניים. בודק את התמקדות קר

ירוחים אられて בעיה של ישראל כל בתו הבירה בספרד. כל התריםihatית של ספרד

ובית

למענה התארך בגרון להפוך בין המונחי "פרדרי" מכון leo עם הדרי

ברוח תושה ונדח סמידה לע שיקוף בינדר לשלוטה עדכני 상 oran התוכניות התכליתות.
A summary of sentiment extraction, which is the process of extracting meaningful words from text in a natural language (English, Hebrew, French, and so on). This means that the extracted meaning can be used to solve a specific task, which is a basic task in the field of natural language processing. This task is multi-faceted, as it can be used in various applications.

Methods based on patterns for solving sentiment extraction problems are methods that use specific patterns to extract information from text. For example, the pattern indicates that the word does not belong to the semantic category of countries. In methods based on patterns, we can search for specific patterns that indicate a specific semantic characteristic in the context of the text. For example, if we look for words such as “countries” in a large text collection, we can use them to extract words that are not found in the text. In turn, texts containing these words are extracted for further analysis.

Methods based on patterns do not have inherent advantages, but an important advantage is the ability to clearly see how the method works. Another advantage is that they can be used for any task that requires sentiment extraction, even if it is not related to sentiment analysis.

Sentiment extraction involves the analysis of text collections that are identified with a specific semantic characteristic. This allows us to search for words that are related to the semantic category of countries using specific patterns. In this case, we can search for words such as “countries” in a large text collection and use them to extract words that are not found in the text. This allows us to extract words that are not found in the text. In turn, texts containing these words are extracted for further analysis.

Methods based on patterns are repeatable, as they are based on specific patterns that can be used to extract information from text. However, they can be computationally expensive and require large amounts of training data.


The summary emphasizes the importance of extracting meaningful words from text in a natural language, which can be used to solve specific tasks in natural language processing.

A summary of sentiment extraction, which is the process of extracting meaningful words from text in a natural language (English, Hebrew, French, and so on). This means that the extracted meaning can be used to solve a specific task, which is a basic task in the field of natural language processing. This task is multi-faceted, as it can be used in various applications.

Methods based on patterns for solving sentiment extraction problems are methods that use specific patterns to extract information from text. For example, the pattern indicates that the word does not belong to the semantic category of countries. In methods based on patterns, we can search for specific patterns that indicate a specific semantic characteristic in the context of the text. For example, if we look for words such as “countries” in a large text collection, we can use them to extract words that are not found in the text. In turn, texts containing these words are extracted for further analysis.

Methods based on patterns do not have inherent advantages, but an important advantage is the ability to clearly see how the method works. Another advantage is that they can be used for any task that requires sentiment extraction, even if it is not related to sentiment analysis.

Sentiment extraction involves the analysis of text collections that are identified with a specific semantic characteristic. This allows us to search for words that are related to the semantic category of countries using specific patterns. In this case, we can search for words such as “countries” in a large text collection and use them to extract words that are not found in the text. In turn, texts containing these words are extracted for further analysis.

Methods based on patterns are repeatable, as they are based on specific patterns that can be used to extract information from text. However, they can be computationally expensive and require large amounts of training data.


The summary emphasizes the importance of extracting meaningful words from text in a natural language, which can be used to solve specific tasks in natural language processing.

חלק מהنتائج שהוצגו בתכנית הבחינה הוצגו במאמר במוסף על כותב מחברת ושותפים למחקר
במהלך תconciliationים של מחברת.


תודות

ברצוני להודות להנחייתם של פרופסורים יואב Goldberg ופרופסורים שואל.Markovitch, ישראל מרקוביץ', על הדרכה והה данיה במהלך תconciliationים של עבורה ו.

אני מודע לכל אלה על תoneksi התוכנית הנérique בהשכלה.

Technion - Computer Science Department - M.Sc. Thesis MSC-2020-42 - 2020
שילוב שיטות מבוססות תבניות ושיטות
מבוססות התפלגות למשימות חילוץ
希מלן

ת NEC על מחקר

לשמ מאייל חלכי של дирישות לקבלת תואר
מגיסר לסי市政府 בית המנהל

גיא קוסלביץ

הועש לפנסי הפקここ — 맺וכן סטטוליגי לישראלי
שתוק לש"מק้า חיפה
אוקטובר 2020
,nil הבוססות הבניות וה⇧יתות המבוססות התפלגות למשימות חילוץ מידיימני

גייל קושלביץ

Technion - Computer Science Department - M.Sc. Thesis MSC-2020-42 - 2020