Discriminating Rhetorical Analogies in Social Media

Christoph Lofi
National Inst. of Informatics
2-1-2 Hitotsubashi,
Chiyoda-ku, Tokyo
101-8430, Japan
lofi@nii.ac.jp

Christian Nieke
TU Braunschweig
Mühlenpfordtstr. 23
38106 Braunschweig
Germany
nieke@ifis.cs.tu-bs.de

Nigel Collier
National Inst. of Informatics
2-1-2 Hitotsubashi,
Chiyoda-ku, Tokyo
101-8430, Japan
collier@nii.ac.jp

Abstract

Analogies are considered to be one of the core concepts of human cognition and communication, and are very efficient at encoding complex information in a natural fashion. However, computational approaches towards large-scale analysis of the semantics of analogies are hampered by the lack of suitable corpora with real-life example of analogies. In this paper we therefore propose a workflow for discriminating and extracting natural-language analogy statements from the Web, focusing on analogies between locations mined from travel reports, blogs, and the Social Web. For realizing this goal, we employ feature-rich supervised learning models which we extensively evaluate. We also showcase a crowd-supported workflow for building a suitable Gold dataset used for this purpose. The resulting system is able to successfully learn to identify analogies to a high degree of accuracy (F-Score 0.9) by using a high-dimensional subsequence feature space.

1 Introduction

Analogies are one of the core concepts of human cognition (Hofstadter, 2001), and it has been suggested that analogical inference is the "thing that makes us smart" (Gentner, 2003). An analogy can be seen as a pattern of speech leading to a cognitive process that transfers some high-level meaning from one particular subject (often called the *analogue* or the *source*) to another subject, usually called the *target*. When using analogies, one emphasizes that the "*essence"* of source and target is similar, i.e. their most discriminating and prototypical processes and properties are perceived in a similar way.

The nature of analogies has been discussed and studied since the ancient Greeks, however computational approaches are still rather limited and in their infancy. One reason for this is that text corpora containing analogies are crucial to study the syntactic and semantic patterns of analogies in order to make progress on automated understanding techniques. For example, to learn about their distribution and the attribute-value pairs that are compared. However, to the best of our knowledge, no such corpus is freely available. We will therefore in this paper present a method for creating such a corpus in an efficient fashion, and make our corpus available for further research efforts.

As an example, consider this brief statement: "West Shinjuku (a Tokyo district) is like Lower Manhattan" It allows readers who know New York, but not Tokyo, to infer some of the more significant properties of the unknown district (e.g., it is an important business district, hosts the headquarters of many companies, features many skyscrapers, etc.). However, automatically understanding analogies is surprisingly hard due to the extensive domain knowledge required in order to perform analogical reasoning. For example, an *analogy repository* containing such domain knowledge has to provide information on which attributes of source and target are generally considered comparable. In contrast to Linked Open Data or typical ontologies, such analogical knowledge is *consensual*, i.e. there is no undisputable truth to analogical information, but a statement can be considered "good" analogical knowledge if its' semantics are perceived similarly by enough people (Lofi & Nieke, 2013). For example, while many properties of West Shinjuku and Lower Manhattan are dissimilar, nonetheless most people will immediately recognize dominant similarities.

In order to build an analogy repositories, a large number of actual analogy statements reflecting the diversity of people's opinions are required for analysis. In this paper, we make a start on this task by proposing a workflow for reliably extracting such statements by using feature-rich supervised learning models, and demonstrate its effectiveness for analogies between different places. Our contributions in this paper are as follows:

- First, we build a suitable *Gold corpus* for training and testing supervised learning models, focusing on *analogies between places*. This corpus will be based upon content mined from search engines and social media.
- We show the effectiveness, but also the challenges of *crowd-sourcing* as a technique for screening and refining potential Gold corpus documents. This process results in multi-sentence text snippets containing an analogy extracted from these documents.
- We design and evaluate supervised learning models with rich feature sets to *recognize analogy statements automatically*, allowing us to substitute crowd-sourcing with automated techniques for further expanding the corpus.
- We extensively *evaluate* our models, and discuss their strengths and shortcomings.

2 Processing Analogies

There exist several approaches for modeling and capturing the semantics of analogies, among them many formal ones relying, for example, on structural mapping (Gentner, 1983). These types of approaches aim at mapping characteristics and relationships of source and target, usually relying on factual domain knowledge given in propositional networks. One example typically used in this context is the Rutherford analogy "Atoms are like the Solar System", which can be derived by outlining similarities between the nucleus and the sun, which are both heavy masses in the center of their respective system, and electrons and planets, which revolve around the center attracted by a strong force (here, the coulomb force is analog to the gravitational force). This model resulted in several theoretical computational models (e.g. (Gentner & Gunn, 2001)).

The most extensively researched subset of analogies are 4-term analogies between two word pairs (mason, stone)::(carpenter, wood). Here, processing analogies boils down to measuring the *relational similarity* of the word pairs, i.e. a mason *works with* stone as a carpenter *works with* wood.

However, measuring the similarity between entities or relationships is a difficult task. While most structure-mapping approaches rely on processing facts, e.g. as extracted from ontologies or knowledge networks, supporters of *perceptual analogies* claim that this similarity has to be measured on a high perceptional level (Chalmers, French, & Hofstadter, 1992), i.e. there can be an analogy if people perceive or believe relations or properties to be as similar even if there are no hard facts supporting it (Kant, 1790), or even when facts oppose it. More formally, two entities A and B can be seen as being analogous (written as $A :: B$) when their relevant relationships and properties are *perceived sufficiently similar* (Lofi & Nieke, 2013). This type of *consensual* analogy is of high relevance in natural communication (in fact, most analogies we discovered in our data are of this type), but very hard to learn as there are no corpora for studying analogies readily available. Furthermore, this definition opens up other challenges: What are the relevant characteristics between two entities? When are they perceived as being similar? And when does an analogy hold true?

With this work, we aim at paving the way for future research on this challenging set of problems by providing a workflow for mining analogy examples from the Web and Social Media. To illustrate this, consider the following example extracted from our Gold corpus:

> "Tokyo, like Disneyland, is sterile. It's too clean and really safe, which are admirable traits, but also unrealistic. Tokyo is like a bubble where people can live their lives in a very naive and enchanted way because real problems do not exist."
> (No. 5310 in corpus)

This perceptual analogy between Tokyo and Disneyland is hard to explain when only relying on typical structured knowledge like Linked Open Data or ontologies, and thus requires specialized data repositories which can be built up using real-world examples as provided by our approach.

Unfortunately, actually detecting the use of an analogies in natural text, a requirement for building sufficiently large test corpora, is not an easy task, as there are only subtle syntactic and morphological differences between an analogy and a simple comparison. These differences cannot be grasped by simple classification models. For example, while many rhetorical analogies contain phrases as "is like" or ", like", as for example in "West Shinjuku is like Lower Manhattan" or "Tokyo is like Disneyland as it is very sterile" there is a plethora of very similar sentences which do not express an analogy ("Shinjuku is like this: …" or "Tokyo, like the rest of Japan, …"). These subtle differences, which are hard to grasp with handcrafted patterns and are often found in the surrounding context, can be modeled by our approach as outlined in section 5.

3 Related Work

There exist several works on the semantics of analogies from a cognitive, philosophical, or linguistic perspective, such as (Dedre Gentner, Keith J. Holyoak, & Boicho N. Kokinov, 2001), (Itkonen, 2005), or (Shelley, 2003).
Hearst-like patterns (Hearst, 1992), which we use as a first and very crude filter during the construction of the Gold dataset, have frequently been employed in recent years, especially in the area of extracting hyponyms, e.g., (Snow, Jurafsky, & Ng, 2004) which also aims at learning new extraction patterns based on word dependency trees. But also approaches for dealing with analogies are frequently based on patterns applied to text corpora. Most of these approaches are tailored for solving general analogy challenges given in a 4-term multiple-choice format, and are usually evaluated on the US-based SAT challenge dataset (part of the standardized aptitude test for college admission). SAT challenges are in 4-term analogy form, e.g. "ostrich is to bird AS a) cub is to bear OR b) lion is to cat", and the focus of those approaches is on heuristically assessing similarity of two given words pairs, to find the statistically more plausible answer. For example, (Bollegala, Matsuo, & Ishizuka, 2009), (Nakov & Hearst, 2008), or (Turney, 2008) approach this challenge by using pattern-based Web search and subsequent analysis of the resulting snippets. In contrast to these approaches, we do not focus on word pair similarity, but given one entity, we aim at finding other entities which are seen as analogous in a specific domain (in our case analogies between locations and places). Being focused on a special domain often renders approaches relying on thesauri like WordNet or CoreLex unusable, as many of the words relevant to the domain are simply not contained.
Closely related to analogy processing is the detection of metaphors or metonyms, which are a special form of analogy. Simplified, a metaphor is an analogy between two entities with the additional semantics that one entity can substitute the other and vice versa). While early approaches to metaphor identification relied on hand-crafted patterns (Wilks, 1978), newer ones therefore heavily exploit the interchangeability of the entities (Beust, Ferrari, & Perlerin, 2003) or (Shutova, 2010), and cannot be used for general analogy processing without extensive adoption. These approaches often also rely on some reasoning techniques based on thesauri, but also other approaches based on mining and corpus analysis became popular. For example in (Shutova, Sun, & Korhonen, 2010) a system is presented which, starting from a small seed set of manually annotated metaphorical expressions, is capable of harvesting a large number of metaphors of similar syntactic structure from a corpus.
Detecting analogies also has some similarities with relation extraction, e.g. (Bunescu & Mooney, 2006) using Subsequence Kernels. However, the task is slightly more difficult than simply mining for a "similar_to" relation, which is addressed by our approach in section 5.

4 Building the Gold Dataset

As the goal of this paper is to supply the tools for creating a large corpus of analogies from the Web, we require a reliable mechanism for automatically classifying if a text snippet contains an analogy or not. Such classification requires a Gold dataset which we construct in this section and which we make available to the community for download[1].
As we expect the number of analogies in a completely random collection of web documents to be extremely low, we first start by collecting a set of web documents that are likely to contain an analogy by applying some easy-to-implement but rather coarse techniques as follows:
In order to obtain a varied set of text snippets (i.e. short excerpts from larger Web documents), we first used a Web search engine (Google Search API) with simple Hearst-like patterns for crawling potentially relevant websites. These patterns were selected manually based on analysis of sample Web data by three experts. In contrast to other approaches relying on extraction patters, e.g. (Turney, 2008) or (Bollegala et al., 2009), our patterns are semi-open, e.g. "# * similar to * as", where # is replaced by one of 19 major cities we used for corpus extraction. * is a wildcard, therefore only one entity of the analogy is fixed by the pattern. Each pattern is created by combining one base part (in this case, "# * similar to *") with an extension part ("as"). We used 17 *different base* parts, and 14 different extensions, resulting in 238 different extraction patterns before inserting the city names. Using Web search, we initially obtained 109,121 search results and used them to crawl 22,360 documents, for which we extracted the text snippets surrounding the occurrence of the pattern (2 preceding and 2 succeeding sentences). The intention of our open Hearst-like patterns is to obtain a wide

[1] http://data.l3s.de/dataset/analogy-text-snippets

variety of text snippets which are not limited to simple analogy cases, so most snippets obtained will actually not be analogies at all. Therefore, additional filtering is required to find those which do actually contain an analogy between places. Unlike e.g. (Turney, 2008) where patterns of the form "[0..1] X [0..3] Y [0..1]", with X and Y two given entities, are used, we chose a more general approach and filtered out all snippets not containing at least two different locations (and hence no place analogy, locations provided by Stanford CoreNLP NER tagger), which left 14,141 snippets.
Since we lacked the means to manually classify all of these snippets as a Gold set, we randomly selected a subset of 8000 snippets, and performed a crowd-sourcing based filtering to detect potential analogies, as described in the following.

4.1 Crowd-Sourcing-Based Filtering

Under certain circumstances, crowd-sourcing can be very effective for handling large tasks requiring human intelligence without relying on expensive experts. In contrast to using expert annotators, crowd-workers are readily and cheaply available even for ad-hoc tasks. In this paper, we used micro-task crowd-sourcing, i.e. a central platform like for example Amazon Mechanical Turk[2] or CrowdFlower[3] assigns small tasks (called HITs, human-intelligence tasks) to workers for monetary compensation. HITs usually consist of multiple work units taking only a few minutes to process, and therefore pay few cents.
Crowd-sourcing has been shown to be effective for language processing related tasks, e.g. in (Snow, O'Connor, Jurafsky, & Ng, 2008) it was used to annotate text corpora, and the authors found that for this task, the combination of three crowd judgments roughly provides the quality of one expert worker. However, the quality can vary due to potential incompetence and maliciousness of workers, making quality control mandatory. The two basic tools for quality control in crowd-sourcing are majority votes and Gold units, which are both used in our process. Gold units are tasks for which the correct answer is known, and they are transparently mixed into normal HITs distributed to workers. If workers repeatedly provide an incorrect judgment for gold units, they are considered malicious, are not paid, and their judgments are excluded from the results.
Therefore, we continued to classify the selected 8,000 snippets using 90 gold units. 5 snippets are grouped within each HIT, for which we pay USD $0.04. For each snippet, 3 judgments are elicited. In total, 336 workers participated in categorizing 87 snippets on average (some top contributors categorized up to 1,975 snippets). As a result 895 snippets are classified as containing an analogy with a confidence of over 90% (confidence is computed as a weighted majority vote of worker judgments and worker reliability; with worker reliability resulting from workers failing or passing gold units in previous tasks).
A brief manual inspection showed that these results cannot be trusted blindly (a correctness of 78% compared to an expert judgment was measured in a small sample), so we performed an expert inspection on all potential analogy snippets, revising the crowd judgments where necessary. Furthermore, we manually tagged the names of the analogous locations. This resulted in 542 snippets which are now manually judged as analogies and 353 snippets that were manually judged as not being an analogy. For this task, worker performance is extremely asymmetrical as it is much easier for crowd-workers to reach an agreement for negative examples than for positive ones, and there were 3,023 snippets classified as no analogies with 100% confidence. This intuition was supported by a short evaluation in which we sampled 314 (10.3%) random snippets from this set and found none that had been misclassified. Therefore, the negative examples of our Gold set consist of the snippets manually re-classified by our expert annotators, and the snippets which had been classified with 100% confidence by the crowd-workers. This leaves out 4,082 snippets for which no clear consensus could be reached, and which are thus excluded from the Gold set.

5 Classifiers and Feature Extraction

Using crowd-sourcing for finding analogy statements is a tedious and still quite expensive task. Therefore, we aim at automating the processes of detecting analogies in a given text snippet by designing multiple rich feature sets for machine learning-based classification models, allowing us to discover new analogies quicker and cheaper.

5.1 Dataset Description

Our complete Gold dataset of 3,918 text snippets shows a ratio of positive to negative examples of roughly 1:8. For training and evaluation, we perform four stratified random selections on the Gold set to obtain 4 training sets with 2/3 of the overall

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

[3] http://crowdflower.com/

size (2,611), and respective test sets with 1/3 size (1,307). In each set, the original ratio between positive example (analogies) and negative examples (not analogies) is retained. We prefer this approach over n-fold cross-validation as some of our models are expensive to train.
All snippets in the Gold set consist of 5 sentences, with 105 words per snippet on average. This average does not significantly vary between positively and negatively classified snippets (94 vs. 106). The overall vocabulary contains 31,878 unique words, with 6,960 words in the positive and 30,234 in the negative subset. 5,316 of these words are shared between both sets (76% of those in the Gold set). This observation implies that the language in our snippets is highly varied and far from saturated (for the significantly smaller positive set, 12.84 new words per snippet are added to the vocabulary on average, while for the larger negative subset, this value only drops to 8.95). This situation looks similar for locations, which play a central role in this classification task: the overall number of different locations encountered in all snippets is 2,631, with 0.86 new locations per snippet in the positive set and 0.73 in the negative set. On average, there are 3.18 locations mentioned in a given snippet, again with no significant differences in the positive and negative subset (3.67 vs. 3.10). Please refer to Table 1 for exhaustive statistics.

5.2 Unigram (Bag-of-Word) Feature Model

As our evaluation baseline, we use a straight-forward unigram (bag-of-word) feature model for training a support vector machine. No stop words are removed, and the feature vectors are normalized to the average length of training snippets. Furthermore, we only retain the 5000 most frequent features, and skip any which occur only in a single snippet. For this experiments (and all other later experiments using a SVM), we used the LibSVM implementation (Chang & Lin, 2011) with a linear kernel due to the size of the feature space.

5.3 N-Gram-based Feature Model

Our first approach to increasing classification quality of the baseline is expanding the feature space to also include n-grams. We tested different versions of this model with *lexical* word-level n-grams, *part-of-speech* n-grams, and *both* of them simultaneously. In all cases, we include n-grams with a length of 1 to 4 words, and similar to the baseline, the top-5000 features are retained and values are normalized to the training snippet length, with a minimal frequency of 2. The required part-of-speech labels are obtained by using the Stanford CoreNLP library[4].The three resulting feature models have been trained and evaluated with three classification algorithms which are known to provide good performance in similar classification tasks: a *support vector machine* classifier (as described in 5.2), a *Naïve Bayes* classifier (provided by the Weka library[5]), and Weka's *J48* implementation of the C4.5 classifier (Quinlan, 1993) (with pruning confidence 0.25 and min. leaf distance 2).

Table 1: Characteristics of Gold Data

characteristic	*all*	*positive*	*negative*
# of snippets	3,918	542	3,376
# of snippets in training set	2,611	361	2,250
# of snippets in test set	1,307	181	1,126
vocabulary size	31,878	6,960	30,234
voc. / #snippets	8.14	12.84	8.95
location vocabulary size	2,631	468	2,459
loc.voc. / #snipts.	0.67	0.86	0.73
# words / s. +	105	94	106
# locations / s.	3.18	3.67	3.10

+ #/s.: average count per snippet

5.4 Shortest Path Feature Model

In this subsection we design the Shortest Path feature model, a model aiming at exploiting some of the specific properties of place analogies. By definition, only text snippets featuring two different places can be a place analogy. The Shortest Path model furthermore assumes that both these locations occur in a single sentence (which is tested in 6.3), and that there is a meaningful lexical or grammatical dependency between these occurrences. For actually building our feature space, we rely on typed dependency parses (Marneffe, MacCartney, & Manning, 2006) of the snippets, and extract the shortest path in the resulting dependency tree between both locations (also using Stanford CoreNLP). This path represents the collapsed and propagated dependencies between both locations, i.e. basic tokens as "on" or "by" are integrated in the edge labels and don't appear as nodes. We considered three variations of this approach: paths built using *lexical* labels, path with *part-of-speech* labels, and a combination of *both*. During the construction of our Gold set, we manually annotated the two relevant places for all

[4] http://nlp.stanford.edu/software/corenlp.shtml

[5] http://www.cs.waikato.ac.nz/ml/weka/

analogies. Therefore this approach can be applied directly for positive training examples. However, for negative snippets, no relevant locations have been annotated. Hence, for all negative snippets in training and all snippets in the test set, we assume that all locations which appear in a snippet (as determined by a NER tagger) are relevant, and we extract all shortest paths between any of them. On average this results in 5.6 paths extracted from any given snippet. The extracted paths are generalized by replacing the locations with a generic, and the final feature model results from constructing a binary feature representing whether a given path occurs or not.
As with the n-gram-based feature model, we train and evaluate SVM, Naïve Bayes, and J48 classifiers with our feature vector (parameters as in 5.3). Please note that building this model is computationally significantly more expensive than the n-gram-based approach as it requires named entity recognition, and typed dependency parsing (we required roughly 30 minutes per training / test set on our Intel i7 laptop).

5.5 Subsequence Pattern Feature Model

Basically, this approach aims at creating something similar to the most common sub forests of all snippets, or skip-grams (Guthrie, Allison, Liu, Guthrie, & Wilks, 2006), i.e. results can be seen as a hybrid between "tree patterns" (as e.g. the Shortest Path) and n-grams. The intention is to avoid the problem of overly local patterns, allowing the patterns to work even in the presence of fill words and subsequences added to a sentence. For this, we utilize the PrefixSpan algorithm (Pei et al., 2001) to detect common, reappearing subsequence in the training set, i.e. sequences of words that appear in a given order, ignoring anything in-between. In contrast to the shortest path approach, this model focuses on multiple sentences simultaneously, and therefore is a significant contribution over state-of-the-art techniques.
As before, we used *lexical*, *part-of-speech*, and *combined* features. The general idea of this approach is to use the PrefixSpan algorithm to mine subsequence patterns from positive gold snippets (the *primitives*), and use these as binary features in a classification step, for which we trained three classifiers as described in 5.3.
In case of the lexical labels, we use the PrefixSpan algorithm to return all subsequences that appear at least 10 times (this value is dependent on characteristics of the dataset and has to be tuned manually) in the relevant part (i.e. the minimal set of consecutive sentences that include both locations)

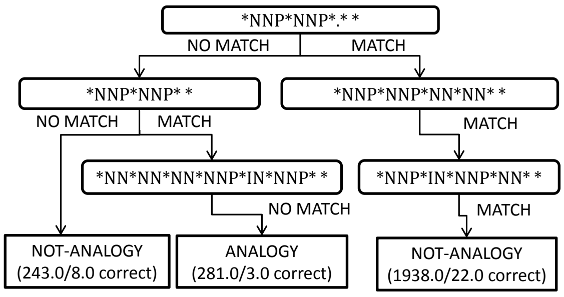

Figure 1: Example Classification Tree

of the positive training set snippets. Depending on the training set used, this resulted in about 40k common subsequences. To avoid unspecific patterns, we filtered out all sequences that did not contain both locations, which reduces the number to about 15k in average. We then replaced the actual locations with a generic, which allows building a regular expression from the pattern that allows any number of words in-between each part of the mined sequence. Before applying a pattern to an unknown snippet, we also replace all (NER tagged) locations with a generic. For example, "LOCATION * is * like * LOCATION" would match "Tokyo is also a lot like Seoul" using regular expressions.
The part-of-speech version is similar to the lexical one, but tries to create more generic and open patterns by mining subsequences from the POS representation of the relevant snippet part. For filtering, all patterns that do not contain two 'NNP' tags and appear less than 60 times are removed (the filter threshold is increased as POS patterns are more generic). We get around 60k to 80k patterns before, and ~10k to 20k primitive patterns after filtering which are used as binary features. Finally, we merged lexical and POS patterns and thus allowed the classifiers to use any of the features. A strongly truncated version of a rule tree created using J48 classification with POS subsequence primitives is shown in Figure 1. Please note that due to the open nature of the primitives and their independence, combining several of them in a feature vector will create extremely complex patterns quite easily. Even a vector that contains only the patterns *A*B* and *A*C* would create matches for ABC, ACB, ABAC, ACAB, AACB, AABC and allow any kind of additional fill words in between. However, this approaches is expensive computationally (testing/training was around 6 hours on average).

6 Evaluation

In the following we evaluate the effectiveness of our analogy classifiers and models. We primarily rely on the *informedness* measure (Powers, 2007) for quantifying performance. In contrast to using only precision, recall, or F-Measure, it respects all error types, false positives (FP) and false negatives (FN), but also true positives (TP) and true negatives (TN), making it a fair and unbiased measure for classification. Furthermore, it compensates biased class distributions in datasets, e.g. as in our dataset the ratio of positive to negative snippets is 1:8, even an "always no" classifier has a correctness of 85%, but will have an informedness of 0. Informedness is given by:

$$informedness = recall + invRecall - 1$$

with: $recall = \frac{TP}{TP+FN}$ and $inverseRecall = \frac{TN}{TN+FP}$

In Table 2, we provide the average informedness, the percentage of correctly classified snippets, F-measure, precision, recall, and inverse recall (true negative rate) for all experiments. A discussion of these results follows in the next section.

6.1 Classifier Performance

Our straight-forward baseline approach, using unigrams and an SVM classifier results in a reasonable informedness of 0.5. Expanding the feature space to lexical n-grams slightly increases performance, while using more generic part-of-speech n-grams results in weaker results. Combining both, however, generally leads to better classification results. When comparing different classification algorithms, it shows that SVMs are most informed when classifying n-grams-based features, followed by J48. Both techniques will result in moderate recall values around 0.5 and precision around 0.6, with a rather high true negative (inv. Recall rate) of 0.9. This changes quite significantly for Naïve Bayes, which is more likely to classify a snippet as positive, therefore leading to higher recall values, but also much lower informedness, precision, and inverse recall. Consequently, the best approach is using SVM with a lexical-POS combined feature space, leading to an informedness of 0.55.

Shortest Path was intended to achieve higher precision results by exploiting additional semantic knowledge of the underlying problem. Unfortunately, it performs poorly if not used with a SVM, but even then it achieves inferior overall results than the best n-gram approach (informedness 0.4). This is due to some of its necessary assumptions not holding true (see section 6.4).

In contrast, our subsequence-based model achieves a higher informedness score of 0.85 and 0.87 in the best cases. While the lexical variants perform not as well, the more generic variants using POS allow for reliable classification. Combining the lexical and the POS features does unfortunately not increase the performance further (quite contrary, the scores generally decrease for combined features). A possible explanation is overfitting caused by the increased feature space.

Table 2: Classifier Result Comparison with respect to the Gold classification

Classifier		Informed.	% Correct	F-Measure	Precision	Recall	Inv. Recall
Always No		*0.00*	*0.85*	*-*	*-*	*0*	*1*
Unigram Lexical	*SVM*	*0.50*	*0.88*	*0.59*	*0.63*	*0.55*	*0.94*
n-Gram Lexical	SVM	0.53	0.89	0.63	0.68	0.58	0.95
n-Gram POS	SVM	0.42	0.87	0.52	0.58	0.48	0.94
n-Gram Lex & POS	**SVM**	**0.55**	**0.90**	**0.65**	**0.73**	**0.59**	**0.96**
n-Gram Lexical	Naïve Bayes	0.33	0.48	0.36	0.22	0.93	0.41
n-Gram POS	Naïve Bayes	0.38	0.61	0.39	0.26	0.81	0.58
n-Gram Lex & POS	Naïve Bayes	0.48	0.75	0.47	0.35	0.73	0.75
n-Gram Lexical	J48 (C4.5)	0.45	0.87	0.55	0.58	0.52	0.93
n-Gram POS	J48 (C4.5)	0.37	0.85	0.47	0.51	0.45	0.92
n-Gram Lex & POS	J48 (C4.5)	0.44	0.87	0.54	0.57	0.52	0.93
Shortest Path	**SVM**	**0.40**	**0.90**	**0.53**	**0.71**	**0.43**	**0.97**
Shortest Path	Naïve Bayes	0.27	0.87	0.40	0.55	0.32	0.96
Shortest Path	J48 (C4.5)	0.26	0.89	0.40	0.77	0.27	0.99
Subseq. Lexical	SVM	0.39	0.87	0.24	0.51	0.46	0.94
Subseq. Lexical	Naïve Bayes	0.53	0.79	0.49	0.36	0.73	0.80
Subseq. Lexical	J48 (C4.5)	0.34	0.86	0.44	0.47	0.41	0.93
Subseq. POS	SVM	0.84	0.97	0.87	0.89	0.85	0.98
Subseq. POS	Naïve Bayes	0.72	0.81	0.57	0.41	0.93	0.79
Subseq. POS	**J48 (C4.5)**	**0.85**	**0.97**	**0.90**	**0.93**	**0.86**	**0.99**
Subseq. Lex & POS	SVM	0.77	0.95	0.83	0.87	0.79	0.98
Subseq. Lex & POS	Naïve Bayes	0.70	0.80	0.56	0.41	0.91	0.79
Subseq. Lex & POS	**J48 (C4.5)**	**0.87**	**0.97**	**0.90**	**0.92**	**0.88**	**0.98**

6.2 Significance Tests

As our Gold set is of limited size, we performed statistical tests to investigate whether the differences reported in the last subsection are actually significant or result from noise. We used an instance-based test relying on the theory of approximate randomization (Noreen, 1989)[6] to perform 100k iterations of randomized testing of the hypothesis that the pairwise performance differences of selected approaches are actually significant (excluding those pairs where the significance is obvious). First, we compared our baseline, lexical unigrams with SVM to using lexical n-grams to test whether using n-grams actually contributed to the quality, and found the difference to be significant (sign-test p<0.024). However, for SVM-based classification, the higher reported performance for also including POS features in addition to lexical n-grams could not be shown to be statistically significant (p>0.4). In the last set of tests, we tested if the choice of the classifier algorithm, SVM or J48, for our two best subsequence-based approaches is significant, and confirmed this clearly (sign-test: p<0.006). According to the reported subsequence results, combining lexical features with part-of-speech features counter-intuitively lowers the performance when using SVM or Naïve Bayes and the positive effect on J48 was shown to be insignificant (p>0.68). Therefore, we can assume that lexical features do not make a substantial contribution when POS features are present.

6.3 Error Analysis

For only 2,845 of all 3,918 snippets, two different locations (regardless of their relevance to the analogy) are mentioned in the same sentence. This severely hampers the effectiveness of our *Shortest Path* approach, which is limited to cases where both locations appear in the same sentence. Those snippets (344 on average / test set) are then classified as "not analogy", decreasing the recall. The overall impact of this shortcoming is still low, as only 4% of these snippets are analogies. Our other approaches are unaffected.
Interestingly, we see what one might call the "inverse problem" when using the other two models (n-gram and subsequence) that search for the presence of certain terms or sequences, but do not explicitly connect them to the locations. They tend to create false positives by detecting statements that contain 2 locations and an analogy, but not between these locations. Consider:

"They say New York is the City of Dreams. I say London is the theatre where it all happens" (No. 5627 in corpus).

Another source for false positives is when an analogy is not stated, but is requested:

"What districts of Paris are similar to Shepherd's Bush or Ealing (both in West London…" (No. 8505 in corpus)

7 Summary and Outlook

We demonstrated approaches for discriminating analogy statements from the Web and Social Media. Our two major contributions are:
a) We created a Gold dataset containing 3,918 example text snippets, of which 542 are positively identified as analogies. This dataset was extracted from 109k potential documents resulting from a Web search with manually crafted Hearst-like patterns. The dataset was consequently refined by using a combination of filters, crowd-sourcing, and expert judgments. We also discussed the challenges arising from a crowd-sourcing in such a setting.
b) Using the Gold dataset, we designed and evaluated a set of machine learning models for classifying text snippets automatically with respect to containing place analogies. Besides more traditional n-gram based models, we also designed novel models relying on feature spaces resulting from shortest path analysis of the typed dependency tree, and high-dimensional feature spaces built from filtered subsequence patterns mined using the PrefixSpan algorithm. In an exhaustive evaluation, the latter approach, which bridges between lexical and structural features, could be shown to provide significantly superior performance with a maximal informedness of 0.87 compared to 0.55 for the next best approach.
In future work, classification performance can be further increased by better handling of current problem cases, e.g. analogies with out-of-domain targets (analogies between locations and other entity classes, analogies between other entities but unrelated locations nearby, etc.) or ambiguous sentence constructions. Also, our approach can be adopted to other domains relevant to Web-based information systems like movies, cars, books, or e-commerce products in general.

[6] Implementation at: http://www.clips.ua.ac.be/scripts/art

However, the more challenging next step is actually analyzing the semantics of the retrieved analogies, i.e. extracting the triggers of why people chose to compare the source and target. Achieving this challenge will allow building analogy repositories containing perceived similarities between entities and is a mandatory building block for actually implementing an analogy-enabled information system.

References

Beust, P., Ferrari, S., & Perlerin, V. (2003). NLP model and tools for detecting and interpreting metaphors in domain-specific corpora. In *Conf. on Corpus Linguistics*. Lancaster, UK.

Bollegala, D. T., Matsuo, Y., & Ishizuka, M. (2009). Measuring the similarity between implicit semantic relations from the web. In *18th Int. Conf. on World Wide Web (WWW)*. Madrid, Spain. doi:10.1145/1526709.1526797

Bunescu, R. C., & Mooney, R. J. (2006). Subsequence Kernels for Relation Extraction. In *Conf. on Advances in Neural Information Processing Systems (NIPS)*. Vancouver, Canada.

Chalmers, D. J., French, R. M., & Hofstadter, D. R. (1992). High-level perception, representation, and analogy: A critique of artificial intelligence methodology. *Journal of Experimental & Theoretical Artificial Intelligence*, *4*(3), 185–211. doi:10.1080/09528139208953747

Chang, C.-C., & Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *2*(3).

Dedre Gentner, Keith J. Holyoak, & Boicho N. Kokinov (Eds.). (2001). *The analogical mind: perspectives from cognitive science* (Vol. 0, p. 541). MIT Press.

Gentner, D. (1983). Structure-mapping: A theoretical framework for analogy. *Cognitive science*, *7*, 155–170.

Gentner, D. (2003). Why We're So Smart. In *Language in Mind: Advances in the Study of Language and Thought* (pp. 195–235). MIT Press.

Gentner, D., & Gunn, V. (2001). Structural alignment facilitates the noticing of differences. *Memory & Cognition*, *29*(4), 565–77.

Guthrie, D., Allison, B., Liu, W., Guthrie, L., & Wilks, Y. (2006). A closer look at skip-gram modelling. In *Int. Conf. on Language Resources and Evaluation (LREC)*. Genoa, Italy.

Hearst, M. A. (1992). Automatic acquisition of hyponyms from large text corpora. In *Int. Conf. on Computational Linguistics (COLING)*. Nantes, France.

Hofstadter, D. R. (2001). Analogy as the Core of Cognition. In *The Analogical Mind* (pp. 499–538).

Itkonen, E. (2005). *Analogy as structure and process: Approaches in linguistics, cognitive psychology and philosophy of science*. John Benjamins Pub Co.

Kant, I. (1790). *Critique of Judgement*.

Lofi, C., & Nieke, C. (2013). Modeling Analogies for Human-Centered Information Systems. In *5th Int. Conf. On Social Informatics (SocInfo)*. Kyoto, Japan.

Marneffe, M.-C. de, MacCartney, B., & Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. In *Int. Conf. on Language Resources and Evaluation (LREC)*. Genoa, Italy.

Nakov, P., & Hearst, M. A. (2008). Solving relational similarity problems using the web as a corpus. In *Proc. of ACL:HLT*. Columbus, USA.

Noreen, E. W. (1989). *Computer-intensive Methods for Testing Hypotheses: An Introduction*. John Wiley & Sons, New York, NY, USA.

Pei, J., Han, J., Mortazavi-asl, B., Pinto, H., Chen, Q., Dayal, U., & Hsu, M. (2001). PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth. *IEEE Computer Society*.

Powers, D. M. W. (2007). Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation. *Flinders University Adelaide Technical Report SIE07001*.

Quinlan, R. (1993). *C4.5: Programs for Machine Learning*. San Mateo, USA: Morgan Kaufmann Publishers, Inc.

Shelley, C. (2003). *Multiple Analogies In Science And Philosophy*. John Benjamins Pub.

Shutova, E. (2010). Models of metaphor in NLP. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Shutova, E., Sun, L., & Korhonen, A. (2010). Metaphor identification using verb and noun clustering. In *Int. Conf. on Computational Linguistics (COLING)*. Beijing, China.

Snow, R., Jurafsky, D., & Ng, A. (2004). Learning syntactic patterns for automatic hypernym discovery. In *Advances in Neural Information Processing Systems (NIPS)*. Vancouver, Canada.

Snow, R., O'Connor, B., Jurafsky, D., & Ng, A. (2008). Cheap and fast---but is it good? Evaluating non-expert annotations for natural language tasks. In *Empirical Methods in Natural Language Processing (EMNLP)*. Honolulu, USA.

Turney, P. (2008). A uniform approach to analogies, synonyms, antonyms, and associations. In *Int. Conf. on Computational Linguistics (COLING)*. Manchester, UK.

Wilks, Y. (1978). Making preferences more active. *Artificial Intelligence*, *11*(3), 197–223.