A case-study of automatic participant labeling

Alexander Kampmann
Saarbrücken Graduate School
of Computer Science
Saarland University
Saarbrücken, Germany
kampmann@st.cs.uni-saarland.de

Stefan Thater and **Manfred Pinkal**
Dept. of Computational Linguistics
Saarland University
Saarbrücken, Germany
stth@coli.uni-saarland.de
pinkal@coli.uni-saarland.de

Abstract

Knowlegde about stereotypical activities like *visiting a restaurant* or *checking in at the airport* is an important component to model text-understanding. We report on a case study of automatically relating texts to scripts representing such stereotypical knowledge. We focus on the subtask of mapping noun phrases in a text to participants in the script. We analyse the effect of various similarity measures and show that substantial positive results can be achieved on this complex task, indicating that the general problem is principally solvable.

1 Introduction

Imagine how you tell a friend about a restaurant visit. You will talk about the people you met there, describe the place and maybe compliment the food. But it is very unlikely that you will tell how the waiter showed you to a table, how you scanned the menu or how the food was brought. Those events are typical for a restaurant. You can assume that the listener knows that they happened, even if you do not mention them.

This kind of knowledge can be captured as a script, a sequence of events which typically constitute a restaurant visit or another common activity (Schank and Abelson, 1977). Scripts also contain information about the participants which take part in an activity.

Script knowledge is implicit in most text sources. This is problematic for text understanding systems. If a text understanding system had access to script knowledge, it could infer the implicit events in the same way a human does. Also the implicit nature of script knowledge means that it is hard to get.

There are approaches to collect script knowledge manually (Mueller, 1998), however, this is too much manual work to be practical. Chambers and Jurafsky (2008) learn narrative schemas from large text corpora. Narrative schemas are similar to scripts, but they are not linked to specific scenarios.

We based our work on the script representations by Regneri et al. (2010) who use crowd-sourcing techniques to collect script knowledge. This approach is scalable and does not suffer from noise, as Chambers and Jurafsky's approach does.

For the script of "Visiting a restaurant", the waiter, the guest and the food are typical participants. Regneri et al. (2011) also processed the script representations to identify participants.

We used texts from the "Dinners from Hell" corpus. This corpus is a collection of texts that were submitted to a website[1] which collects stories about bad experiences with restaurants. The texts feature harsh waitstaff or over-expensive restaurants, which means the Restaurant script is the main topic. Thereby a lot of the script events are mentioned explicitly. The corpus also contains some texts which incorporate deviations from the script. As an example, one text explains how a large chandelier falls on the customer's table. This event is not included in the Restaurant script.

In this paper, we present a system that labels noun phrases with the script participants they refer to. As a case study, we evaluated our system on the "Dinners from hell" corpus. We applied participant knowledge from the Restaurant script that was devised by Regneri et al. (2011). 1 shows a sentence with labels automatically assigned by our system. The participants "customer", "food", "drink", "table" and "location" have been identified. We can also see a mistake in the automated labeling. The tickets are labeled as "bill". This happens, because the Restaurant script is about a restaurant were the customers give a spoken order, rather than a written one.

Our contributions are the following:

[1] www.dinnersfromhell.com

Proceedings of the Int. Conference of the German Society for Computational Linguistics and Language Technology,
pages 97–105, University of Duisburg-Essen, Germany, Sep 30–Oct 2 2015.

We (customer) each ordered entrees (food) and sodas (drink), placed the tickets (bill) on our table (table), and went to the buffet line (location) for some salads (food).

Figure 1: A sentence from the development set with generated annotations.

1. We describe an automated participant labeler, which tries to find the participant that is most similar for each noun phrase. We compare several variants of a model which can be used to calculate similarities between participant mentions in texts and script occurrences of a participant.

2. We evaluate our similarity functions on the "Dinners from hell" text corpus, together with the Restaurant script that was provided by Regneri et al. (2011).

3. Our analysis also provides an assessment of the completeness and usefulness of available script information.

The paper is structured as follows: 2 gives an overview of Regneri et al. (2010) and Regneri et al. (2011), which provided the script representations we used. In 3, we describe the annotation process. 4 describes our participant labeling. In 5, we discuss our results. In 6, we conclude the paper.

2 Script data

We used the script representations from Regneri et al. (2010).

In this work, several descriptions of typical instances of a variety of scripts were collected on Amazon's Mechanical Turk[2]. The Mechanical Turk provides access to "Human Computing Resources". This means that human annotators are asked to solve small tasks and are payed small amounts of money for each task.

Regneri et al. (2010) asked annotators on MTurk to provide descriptions of the typical event sequence for a scenario in a telegram-style. This leads to several "Event Sequence Descriptions" (ESDs). In each ESD, there is one sentence per event, however, different annotators often provided different descriptions for the same event. Also the

[2]http://mturk.com

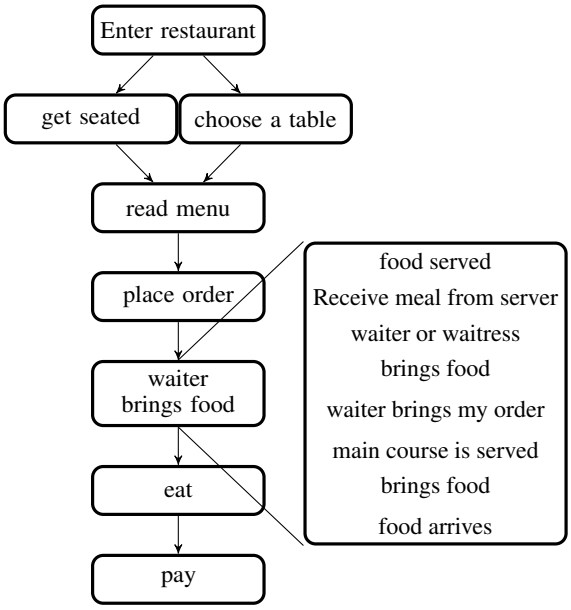

Figure 2: Simplified representation of the restaurant script, with a full view of the "waiter brings food" event.

ESDs from different annotators differ in granularity, and thereby in length. Regneri et al. (2011) aligned event descriptions by a version of multiple sequence alignment which is sensitive to semantic similarity. They obtained a graph representation of the script, where each node contains a set of similar event descriptions, and the edges denote their order in the ESDs. 2 shows a simplified version of the Restaurant script. Each node in the script graph contains all the sentences the annotators provided for this event. In the figure, we labeled each node with a single description. We show the full list for the "waiter brings food" event as an example.

Each noun phrase in an ESD refers to a participant of the script, but different noun phrases may refer to the same participant. In the example, the "waiter" participant is realized as "server" in the second sentence, but as "waiter or waitress" in the third. In follow-up work Regneri et al. (2011) used an Integer Linear Program (ILP) to group the noun phrases to "participant description sets" (PDS). The participant description set for the waiter is $\{\text{waiter}, \text{waitress}, \text{server}\}$

For our experiment, we did not use the ILP approach, but specified a gold standard for the Restaurant script manually. This gold standard is based on noun phrases from Regneri's work, but the grouping has been done manually. We decided to do so, because the ILP solution was to noisy to get good results.

3 Text data: Dinners from Hell

The "Dinners from Hell" corpus has been extracted from the "Dinners from Hell" website. The main topic of the texts in the corpus are restaurant visits, so the script knowledge is explicit in most of them. This makes the texts a good subject for our work.

The corpus contains some texts which describe (displeasing) deviations from the restaurant script. This is a challenge for script-knowledge based systems, as they may not be able to cope with atypical scenarios. On the other hand, there are several texts which have all the typical script events. As an example, one of the stories is about an overly expensive restaurant, another one features a harsh waiter.

We took existing annotations (Rudinger et al., 2015) as a starting point. Three annotators marked independently which verbs they consider relevant for a given script. In the sentence

> I was near Harvard Square and decided to have lunch at a small Chinese restaurant.

the verb "was" would not be marked, because being in a special place is not part of the restaurant script. On the other hand, "have lunch" is typical for a restaurant, so it is marked as script-relevant. The word "decide" was not marked as script-relevant by two annotators, while the third one marked it. In such cases, the solution most annotators used was considered as correct. So "decide" is not script-relevant here.

We parsed the texts with CoreNLP (Manning et al., 2014) and considered direct dependents of event verbs as candidates for participant annotation.

We extended the annotation with participant labels for those candidates.

In the ideal case, the labels in the manual annotation would be the participants that are contained in the script representation. However, it turned out that the script data is incomplete. As an example, none of the ESDs used to derive the script representation contained the option of taking left-overs in a to-go box, which happens regularly in the texts. To account for this problem, we extended the tag set for the annotations. 1 lists the tags that were used in the manual annotation in the left column. The right column contains which of the participant description sets from the Restaurant script we considered equivalent to a gold label.

Gold Annotation	Accepted labels
amount	
coupon	
reservation	
to-go box	
utensils	
customer	customer
cashier	waiter
management	waiter
waiter	waiter
restaurant_institution	waiter
cook	cook
condiments	food
drink	drink
food	food
food_or_drink	food, drink
order	drink, food
credit card slip	credit card slip
bill	bill
tip	tip
kitchen	kitchen, restaurant_location
location	restaurant_location
restaurant_location	restaurant_location
menu	menu
payment method	payment method
table	table

Table 1: Mapping of gold tags to automated labels. Empty fields mean this annotation can not be matched by the participant labeler.

In some cases, the parse trees contained errors. The annotator assigned the '- - -' label to incorrectly recognized words. Words which were direct dependents of an event verb, but missed by the parser, were tagged as well.

During the annotation, it turned out to be difficult to assign unique labels to all participants. As an example, a person who brings food to the table usually is a waiter. However, it may happen that the owner or manager brings food. In this case the owner works as a waiter in his own restaurant. The annotator assigned the waiter label if the person served as a waiter, however, if it was clear from the story that this is the owner, the owner label was used, even if the person took over a waiter's responsibilities.

For this exploratory study, we rely on a single annotator. The manual annotations are used as a gold standard, the evaluation compares the results

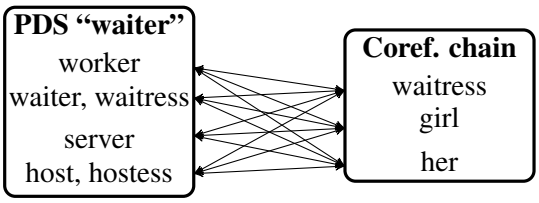

Figure 3: Similarity values between a participant description set (PDS) and a coreference chain.

of our participant labeler to them.

4 Models for Participant Labeling

The automated participant labeler receives the verb annotations in the text and the participant description sets from the script as input. The verb annotations contain the information which verbs are script-relevant. The labeler uses this information and the CoreNLP parser to identify direct dependents of script-relevant verbs as participant candidates. Those will be labeled later on, which is the same thing the annotator did conceptually.

The labeler uses CoreNLP to get coreference chains. We assume that two words in the same coreference chain denote the same participant, so the participant labeler assigns a role label for each coreference chain. In order to do so, it calculates a similarity score between the head nouns of the noun phrases in a coreference chain and the words in a PDS. 3 shows a PDS and a coreference chain.

We obtained several similarity values between each coreference chain and PDS. In 3, three words in the coreference chain and four words in the PDS lead to 12 similarity values. The edges illustrate the twelve similarities we calculate. For the labeling, we need a single similarity value for each participant. In the "max" approach, we calculated the maximum of all similarity values between a coreference chain and a PDS. In the "mean" approach, we used the arithmetic mean of all the values as the similarity. In both cases, all words in a coreference chain are labeled with the participant which reaches the highest similarity value.

4.1 WordNet-based similarity functions

WordNet (Miller, 1995) is a database which contains english words and relations between them. Most notable, WordNet relates hypernyms to their hyponyms. We expect that the script representation contains general terms, while the text rather use the more specific words. This motivates that we use the **hyponym** similarity metric, which comes as a built-in in NLTK (Bird, 2006). We also included **Lin's WordNet-based similarity** (Lin, 1998) and the **Wu-Palmer similarity** (Wu and Palmer, 1994).

All WordNet-based similarity functions calculate a similarity score between two WordNet synsets. SynSets group words with the same meaning together. Words can be in more than one synset, if they have several meanings. As an example, "card" may, among others, refer to the "menu" as well as to a "credit card", so the word is in several synsets.

For our similarity functions, we took all synsets that contain one of the words in the participant and compared them to all the synsets that contain a participant candidate in the respective coreference chain. Again this yields several values per coreference chain. We used an weighted arithmetic mean or simply the maximum to get a single value. The weights in the arithmetic mean are the occurrence counts of words from the participant in the respective synsets.

4.2 Verb-occurrence similarity functions

Some participants occur with some verbs more frequently than others. As an example, "waiters" "bring" food, so the subject of "bring" is most likely a waiter. Thereby we designed a similarity function which uses the verb a participant candidate depends on.

For this similarity function, we worked with the entire coreference chain and the entire PDS directly, rather than comparing individual words. Each participant candidate is a direct dependent of an event-relevant verb. We counted how often which verb is used with a participant candidate in the coreference chain. This gives us a vector of verb occurrence counts.

We also counted the verbs which were used in the PDS together with the noun phrases in the ESDs. This yields a second occurrence count vector. The cosine similarity between the vectors serves as similarity value for the verb similarity function.

4.3 Distributional similarity functions

We complement the knowledge-based WordNet similarity scores with similarity scores from a distributional semantic model. The key idea behind distributional models is that semantically similar words tend to occur in similar linguistic contexts in large text corpora. Distributional models represent these contexts as vectors and compute se-

mantic similarty by comparing these vectors in a high-dimensional vector space.

We use the model of Thater et al. (2011), who train a vector space model from a dependency parsed version of the English Gigaword corpus. We use the model in two ways: In the "uncontextualized" mode we simply compare the vector of the target word with the vector of the word representing the participant in the script; in the "contextualized" mode, we first contextualize the vector of the target word using the syntactic context in which it occurs before comparing it with the vector of the participant. The basic intuition here is that contextualizing a vector should improve the similarity scores for ambiguous target words ("look at *card*" vs. "pay with *card*"). It turns out, however, that the use of uncontextualized vectors leads to a better performance compared to using contextualized vectors (see Section 5).

In cases where the target word occurs in a coreference chain, we used both the sum as well as the mean of the pairwise similarity scores of the vectors of the words in the coreference chain and the vector for the participant.

5 Evaluation

Our approach is not a machine-learning approach. The similarity functions are not trained on our data. However, especially for the combinations of similarity functions, there was a lot of manual optimization needed. In order to avoid a subject bias here, we divided the "Dinners from Hell" into a test and a development set. None of the authors had a look at the test set data until the final evaluation started.

The development set contained 71 texts with 28707 words. The test set contains 72 texts with 28600 words. During the annotation, the annotator reported some problems. Our inspection of the problematic texts lead to updates in the annotation guidelines. All texts we saw in this process were taken to the development set. After that, we selected additional texts for the development set randomly.

We ran our participant labeler on the test set and calculated the precision and recall for a label p as given in 1 and 2.

$$\text{precision}(p) = \frac{|TP_p|}{|TP_p + FP_p|} \quad (1)$$

$$\text{recall}(p) = \frac{|TP_p|}{|TP_p + FN_p|} \quad (2)$$

Intuitively, precision is the fraction of correct labels among the assigned labels. Recall is the fraction of participants which are identified and labeled correctly by the participant labeler.

To give an impression of the performance of the participant labeler with respect to all labels, we use micro and macro averages of precision and recall as given in:

$$\text{macroprecision} = \frac{1}{|P|} \sum_{p \in P} \text{precision}(p) \quad (3)$$

$$\text{macrorecall} = \frac{1}{|P|} \sum_{p \in P} \text{recall}(p) \quad (4)$$

$$\text{microprecision} = \frac{\sum_{p \in P} |TP_p|}{\sum_{p \in P} |TP_p| + |FP_p|} \quad (5)$$

$$\text{microrecall} = \frac{\sum_{p \in P} |TP_p|}{\sum_{p \in P} |TP_p| + |FN_p|} \quad (6)$$

In 3 and 4 all participants have the same influence, no matter how often they occur. 5 and 6 are more relevant for our task, because they consider how often a participant occurs in total, so mistakes in participants which occur seldom have less effect than mistakes in more frequent participants.

The participant labeler and the annotator worked under different conditions. First of all, the annotator fixed parser errors on the fly. Our participant labeler is incapable of doing so. Also the annotator used a tag set that contains some participants that do not occur in the script representation. One example is the "to-go box". None of the initial ESDs contained the option to take left-overs home in a to-go box. 1 lists the tags that the annotator used in the left column, the right column lists the tags our participant labeler uses which we consider equivalent.

One of the main differences is in the "food" tags. The annotator assigned the "food" label for "food" and the "drink" label for drinks. If it was not obvious whether it was drink or food, as in the sentence "the waiter brought our order" the "food_or_drink" tag was used. The participant labeler never uses the "food_or_drink" tag, it always commits to one of the specific options.

This leads to a problem with respect to false negatives. If the annotator assigned the "food_or_drink" tag and the participant labeler decides for something wrong, is this counted as a false negative

Heuristic		Micro-			Macro-		
		Precision	Recall	F-Score	Precision	Recall	F-Score
Base lines							
perfect labeling		0.86	0.86	0.86	0.83	0.78	0.80
manual WordNet		0.02	0.02	0.02	0.00	0.07	0.00
string equality		0.22	0.21	0.21	0.46	0.42	0.44
random		0.06	0.05	0.05	0.05	0.05	0.05
most frequent participant		0.32	0.27	0.29	0.02	0.07	0.03
WordNet-based similarity functions							
Lin	mean	0.43	0.38	0.41	0.33	0.47	0.39
	max	0.36	0.37	0.37	0.28	0.42	0.34
Wu-Palmer	mean	0.40	0.35	0.37	0.29	0.47	0.36
	max	0.37	0.37	0.37	0.35	0.54	0.42
Hypernym	mean	0.51	0.49	0.50	0.38	0.53	0.45
	max	0.59	0.59	0.59	0.42	0.55	0.47
squared lin	mean	0.52	0.49	0.50	0.35	0.52	0.42
	max	0.35	0.34	0.34	0.28	0.41	0.33
squared Wu-Palmer	mean	0.46	0.42	0.44	0.32	0.51	0.39
	max	0.37	0.37	0.37	0.35	0.54	0.42
Verb similarity functions							
verb		0.10	0.08	0.09	0.07	0.16	0.10
Distributional similarity functions							
VSM, sp, max		0.36	0.37	0.36	0.37	0.57	0.45
VSM, cosine, max		0.35	0.36	0.35	0.34	0.53	0.42
VSM, cosine, mean		0.35	0.36	0.36	0.32	0.53	0.40
cosine, max		0.28	0.27	0.28	0.27	0.41	0.32
sp, max		0.29	0.29	0.29	0.28	0.44	0.34
cosine, mean		0.28	0.28	0.28	0.27	0.41	0.33
Combinations of similarity functions							
Wu-Palmer + verb, max		0.33	0.32	0.33	0.25	0.41	0.31
sq. Wu-Palmer + verb, max		0.35	0.35	0.35	0.26	0.45	0.33
Wu-Palmer + Verb, mean		0.38	0.32	0.35	0.21	0.37	0.27
Lin + Verb, mean		0.22	0.18	0.20	0.17	0.32	0.22
mean of Wu-Palmer + max of sq. Lin		0.57	0.56	0.56	0.31	0.56	0.40
mean Wu-Palmer + cosine, max		0.55	0.53	0.54	0.36	0.57	0.44
hill-climbing		0.60	0.60	0.60	0.35	0.59	0.44
sum of all		0.38	0.39	0.38	0.35	0.57	0.44

Table 2: Results of our participant labeling. VSM means the vector space model with no context information. sp denotes the scalar product

with respect for the "food" tag or the "drink" tag? We double counted here, which leads to a higher number of false negatives.

The participant labeler can not automatically correct parsing errors, thus it can not assign the same labels as in the gold standard in all cases. Also the double counting effect obscures the evaluation metrics. To account for those effects, we added a simulated, perfect labeling, which assigns the gold annotation in cases the participant labeler can handle and a wrong tag otherwise. This perfect labeling serves as an upper bound for the performance of our automated labeling.

We added three base lines. String equality counts how many words in a coreference chain are equal to words in the participant and normalizes by the number of words in the chain. The random assignment assigns a participant randomly, most frequent participant assigns "customer", which is the most frequent participant in the "Dinners from hell" corpus, to all participant candidates.

5.1 WordNet-based similarity functions

As a base line for WordNet-based approaches we manually selected a WordNet-synset per participant and labeled each participant candidate with the closest participant according to Lin's similarity measure. The results are given as 'manual' in 2. This approach is substantially outperformed by the automated synset selection. We selected just one synset per participant, but for some participants, there is no single, descriptive synset, which we think is the problem about the manual selection.

For the WordNet-based similarity functions, we get several values per function, because there are several synsets per participant. We used either a weighted average or the maximum to get a single value, so there are two rows per WordNet-based similarity in 2.

For an arithmetic mean, small differences between values can be lost due to the mean calculations. On the development set, we observed that this effect happened for several cases in Lin and Wu-Palmer similarity. To counteract this effect, we also tried squared versions of both similarity measures. Squaring a similarity score makes small values smaller, but has little effect on values close to 1. Thereby squaring the similarity gives larger differences between large values and close-by, but smaller values.

With our WordNet-based approaches, we achieve a Micro-Precision of up to 59%, the Macro Precision reaches 42%. Squaring has a positive effect on the mean similarity and almost no effect on the maximum similarity, which is expected, as squaring does not change which of the numbers is larger.

5.2 Verb occurrence similarity function

The verb occurrence similarity function reaches a Micro-Precision of 10% and a Macro-Precision of 7%. This is substantially worse than the wordNet-based approaches. However, words like "he" or "she" do not occur in WordNet, so the WordNet-based approaches can not label them, unless there is some word that offers more information in the same coref-chain. With the verb similarity function, our participant labeler managed to label some mentions of pronouns correctly.

5.3 Distributional similarity functions

For the distributional similarity functions, we evaluated how context information influenced the decisions and whether it made a difference if we used the maximum or the average to get a single value for a coreference chain. Also we compared the cosine and the scalar product as vector similarity.

The best performing similarity function in this category uses no context information, the scalar product and the maximum. I reaches a Micro-Precision of 36%. All in all, the scalar product seems to be better than the cosine similarity, which confirms earlier results with other distributional approaches.

All in all, the distributional approaches do not outperform the WordNet-based approaches.

5.4 Combining similarity functions

We observed that all of the similarity functions perform well in different situations, so it is reasonable to assume that combinations may perform even better. As a test, we averaged Wu-Palmer and Lin similarity. The result is the second best heuristic in our evaluation, which encouraged us to try several combinations.

As a base line, we averaged all similarity functions, except for the scalar products. The reason to exclude the scalar products is that all other similarities are between 0 and 1, which the scalar product is not. This means the scalar product would dominate all other heuristics. The results for the sum are included in 2 as "sum of all."

Additionally, we experimented with several combinations of individual similarity scores. First, we designed some combinations by hand. Our intuition was that WordNet-based similarities and the verb similarity should play together well, because they provide information for different parts of speech. Second, we combined functions with a high dissimilarity between the confusion matrices, which leads to a selection of functions which make mistakes for different participants. Third, we applied a simulated hill-climbing on the development set to find a good combination. This heuristic is basically a weighted sum of other heuristics.

The results of the combinations are somewhat discouraging. Apparently, the WordNet-based heuristics are overrated by high scores from the verb heuristics. So a well-performing heuristic can not reliably overvote a bad performing one.

The most dissimilar confusion matrices can be found for the distributional model with context information and the mean of Wu-Palmer similarity. The combination outperforms both partners, and almost reaches the best performing individual score. So the idea of combining heuristics with different errors seems to work.

Hill climbing leads to the best performing combined similiarity score in terms of Micro F-Score. However, the differenece to the best performing individual score (Hypernym, max) is small.

6 Conclusion

In this paper, we reported on a case study on automatically mapping noun phrases in texts to participants in scripts, and showed that substantial positive results can be achieved on this complex task. Our automatic participant labelling system uses similarity scores between pairs of words to identify the most apropriate pariticipant for a given noun phrase. We investigate the effect of various similarity measures. The best individual measure achieves a (Micro-) F-score of 0.59, compared to the baseline of 0.29 (most frequent participant) and the upper bound of 0.86. The system performance can be further improved to 0.6 by combing different similarity measures using hill climbing.

The WordNet-based similarity functions perform reasonably well on this task. This came as a surprise to us, as we initially believed it would not contain a lot of relevant words.

We were also surprised that distributional similarity functions can not outperform the WordNet-based functions. We believe that the problem is that our words are all from the same domain (restaurant), and thereby have high (distributional) similarity anyways.

Our results hold for the "Dinners from hell" corpus. It is not clear whether our similarity functions can be applied in domains other than restaurant visits. More texts and different scripts would be necessary in order to evaluate this.

Also we do not believe that we found the best possible combination of similarity functions yet. While hill climbing yielded a good similarity function, it outperforms our initial, uninformed guess only slightly. It might be possible to come up with a better way to combine similarity scores.

The WordNet-based approaches work best on nouns. Distributional approaches, on the other hand, should be able to handle all word classes. Thereby it is a straight-forward idea to implement a similarity function which relies on distributional similarity for everything except nouns, which can be handled by WordNet. We did not implement this approach so far.

However, we got the impression that sparse script data is a more severe problem. Some participants which occur in the texts do not occur in the script. Further research about how to collect script data is required.

Acknowledgments

We thank Alessandra Zarcone and our student assistant Tatiana Anikina for providing the annotation used in our study.

References

Steven Bird. 2006. NLTK: The natural language toolkit. In *Proceedings of the COLING/ACL on Interactive Presentation Sessions*, COLING-ACL '06, pages 69–72, Stroudsburg, PA, USA. Association for Computational Linguistics.

Nathanael Chambers and Daniel Jurafsky. 2008. Unsupervised learning of narrative event chains. In *ACL 2008, Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, June 15-20, 2008, Columbus, Ohio, USA*, pages 789–797.

Dekang Lin. 1998. An Information-Theoretic Definition of Similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 296–304, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

George A. Miller. 1995. WordNet: A Lexical Database for English. *Commun. ACM*, 38(11):39–41, November.

Erik T Mueller. 1998. *Natural language processing with ThoughtTreasure*.

Michaela Regneri, Alexander Koller, and Manfred Pinkal. 2010. Learning script knowledge with web experiments. In *Proceedings of ACL 2010*, Uppsala, Sweden, July. Association for Computational Linguistics.

Michaela Regneri, Alexander Koller, Josef Ruppenhofer, and Manfred Pinkal. 2011. Learning Script Participants from Unlabeled Data. In *Proceedings of RANLP 2011*, Hissar, Bulgaria.

Rachel Rudinger, Vera Demberg, Ashutosh Modi, Benjamin Van Durme, and Manfred Pinkal. 2015. Learning to predict script events from domain-specific text. *Lexical and Computational Semantics (* SEM 2015)*, page 205.

Roger Schank and Robert Abelson. 1977. Scripts plans goals and understanding.

Stefan Thater, Hagen Fürstenau, and Manfred Pinkal. 2011. Word Meaning in Context: A Simple and Effective Vector Model. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1134–1143, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing.

Zhibiao Wu and Martha Palmer. 1994. Verbs Semantics and Lexical Selection. In *Proceedings of the 32Nd Annual Meeting on Association for Computational Linguistics*, ACL '94, pages 133–138, Stroudsburg, PA, USA. Association for Computational Linguistics.