Learning Script Participants from Unlabeled Data

Michaela Regneri * **Alexander Koller** † **Josef Ruppenhofer** ‡ **Manfred Pinkal** *

* Department of Computational Linguistics, Saarland University
{regneri,pinkal}@coli.uni-saarland.de
† Department of Linguistics, University of Potsdam
koller@ling.uni-potsdam.de
‡ Department of Information Science and Language Technology, University of Hildesheim
Josef.Ruppenhofer@uni-hildesheim.de

Abstract

We introduce a system that learns the participants of arbitrary given scripts. This system processes data from web experiments, in which each participant can be realized with different expressions. It computes participants by encoding semantic similarity and global structural information into an Integer Linear Program. An evaluation against a gold standard shows that we significantly outperform two informed baselines.

1 Introduction

Scripts (Schank and Abelson, 1977) represent commonsense knowledge about the events that stereotypically constitute a certain activity. For instance, the "restaurant" script might specify that the patron enters, the waiter shows the patron to their seat, eventually the patron eats a plate of food, and so forth. There has always been agreement that script knowledge can be highly useful for a variety of applications in artificial intelligence and computational linguistics, including commonsense reasoning for text understanding (Cullingford, 1977; Mueller, 2004), information extraction (Rau et al., 1989) and automated storytelling (Swanson and Gordon, 2008). But there is hardly an area where the discrepancy between the felt importance of a type of knowledge and the inability to provide any substantial amount of this knowledge for serious applications is greater.

Recently, several groups have tackled the problem using unsupervised methods for learning script-like knowledge from text corpora or data obtained through web experiments (Chambers and Jurafsky, 2008; Chambers and Jurafsky, 2009; Regneri et al., 2010). For the first time, they open up a perspective to wide-coverage resources of script knowledge. However, each of these approaches handles only specific aspects of script information: Chambers and Jurafsky (2009) learn *narrative schemas* and their participants; they group verbs into schemas by virtue of shared participants assuming that this is an indicator for being part of the same stereotypical activity, without knowing the actual scenarios. The system of Regneri et al. (2010) learns the temporal order of events occurring in specific stereotypical scenarios, but does not determine participants.

In this paper, we present a system that automatically learns sets of participants associated with specific scenarios. We take the approach of Regneri et al. as our starting point. In this earlier work, several experimental subjects described what happens in a given scenario in a web experiment; the system then learns what event descriptions from different subjects refer to the same event, and how they are temporally ordered, using Multiple Sequence Alignment (Durbin et al., 1998). The specific problem we consider is to group the different noun phrases occurring throughout a script into equivalence classes, resulting in one class for each participant. Our solution combines diverse sources of information, including semantic similarity and structural information from the sequence alignment, in an Integer Linear Program (Wolsey, 1998, *ILP*). The desired equivalence classes then correspond to an optimal solution of the ILP. We not only show that our system significantly outperforms a high-precision baseline, but also that it substantially exploits global structural information. The process is almost entirely unsupervised: We rely on annotated data only for training a handful of similarity thresholds and for evaluation. We expect our approach to scale up and help obtain a broad-coverage knowledge base of scripts with participants through web experiments.

Plan of the paper. The paper starts by reviewing related work. We will then define the exact script learning problem we tackle here. Next, we show how participants can be learned, and then present

	ESD 1	*ESD 2*	*ESD3*
1	put food on plate	put food in bowl	put food on dish
2	open microwave	open door	open oven
3	put plate in	put food inside	place dish in oven
4	close microwave	close door	close
5	∅	enter time	select desired length
6	press start	push button	∅
7		...	

Figure 1: Alignment for the MICROWAVE scenario.

the evaluation before we finally conclude.

2 Related Work

Many papers on scripts and their application perspectives have been published in the seventies (Schank and Abelson, 1977; Barr and Feigenbaum, 1981). Script knowledge was manually modeled, and never exceeded a handful of domains and implementations operating on them.

Scenario frames in FrameNet (Baker et al., 1998) are another approach to modeling scripts and their participants. They describe how a stereotypical activity is made up of smaller events (frames), which share roles (frame elements) specifying people and objects involved in the events.

The supervised approach of Mani et al. (2006) learns temporal event relations from TimeBank (Pustejovsky et al., 2006).

All of these approaches rely on elaborate manual annotation efforts, and so it is unclear how they would scale to wide-coverage resources.

Chambers and Jurafsky (2008; 2009) exploit coreference chains and co-occurrence frequency of verbs in text corpora to extract *narrative schemas* describing sequences of events and their participants.[1] Because this approach is fully unsupervised, its coverage is in principle unlimited. Each schema provides a family of verbs and arguments related by the same narrative context. Roughly speaking, event sequences are induced by grouping verbs in the same schema if they tend to share the same arguments. Within the schemas, events are represented as verbs, while the relations between the verbs remain underspecified: Two verbs of a schema might describe the same, different or contradictory events. The aim here is not to collect data describing predetermined activities, but rather to establish verb groups that share an (unknown) underlying scenario.

Regneri et al. (2010) (henceforth, RKP) propose an alternative approach with complementary strengths and weaknesses. The starting point are specific scenarios, and human users answer questions like “what happens in a restaurant?”. From the data collected in this way, a mining algorithm learns both which phrases describe the same subevent and how these sub-events are ordered temporally. This guided way of learning script data produces representations associated with known scenarios, and also opens up the possibility of learning about activities that are too stereotypical to be elaborated much in text corpora (and which thus can’t be induced from there). However, the approach is limited by its reliance on scenarios that have to be determined beforehand. Tying in with this previous work, we compute participants using Integer Linear Programming to globally combine information from diverse sources. ILP has been applied to a variety of different problems in NLP (Althaus et al., 2004; Barzilay and Lapata, 2006; Berant et al., 2010), including coreference resolution (Denis and Baldridge, 2007; Finkel and Manning, 2008).

3 Scripts and Participants

We formalize the problem of computing participants of a script as one of computing equivalence classes of mentions occurring in script-related event descriptions. In this respect our task is similar to coreference resolution.

Our algorithm takes the raw data and processed outputs of RKP as its starting point. The RKP data consist of a collection of *event sequence descriptions (ESDs)*, each of which is written by one annotator to describe how a scenario plays out. RKP compute an *alignment table* out of the ESDs (Fig. 1) using Multiple Sequence Alignment (Durbin et al., 1998, *MSA*). The columns of this alignment table represent the original ESDs, possibly interspersed with some gaps (“∅”). The non-gaps in each row are *aligned*, and thus presumably describe the same event in the scenario (cf. *open microwave*, *open door*, and *open oven* in Fig. 1). The MSA algorithm assumes a *cost function* for substitutions (= aligning two non-gaps) and *gap costs* for aligning gaps with non-gaps to compute the lowest-cost alignment of the ESDs.

In our work, we compute script-specific participants using the alignment tables. For example, we want to find out that *plate*, *bowl* and *dish* fill the same role in the microwave script. We call a mention of a participant (typically a noun phrase) in

[1] See http://cs.stanford.edu/people/nc/schemas

some event description a *participant description*. Our system is intended to group participant descriptions into equivalence classes, which we call *participant description sets* (PDS).

4 Computing Participants

Learning participants from aligned ESDs is done in two steps: First, we identify candidate participant descriptions in event descriptions. Then, we partition the participant descriptions for each scenario into sets. The sets correspond to script-specific participants, their members are possible verbalizations of the respective participants.

4.1 Identifying participant descriptions

We consider participant descriptions to be the noun phrases in our data set, and thus reduce the task of their identification to the task of syntactic parsing. Parsing event descriptions is a challenge because the data is written in telegraphic style (cf. Fig. 1). The subject (typically the protagonist) is frequently left implicit, and nouns lack determiners, as in *start microwave*. In our experiments, we use the Stanford parser (Klein and Manning, 2003). Under the standard model, parsing accuracy for phrase structure trees is only 59% on our data (evaluated on 100 hand-annotated example sentences). The scores for dependency links between predicates and direct objects indicate how many noun phrase heads are correctly identified. Here the standard parser reaches 81% precision. The most frequent and most serious error is misclassification of the phrase-initial verb (like *start*) as a noun, which often leads to subsequent errors in the rest of the phrase.

Our available dataset of event descriptions is much too small to serve as a training corpus of its own. To achieve sufficient parsing accuracy, we combine and modify existing resources to build the parser model: we re-train the parser on a corpus consisting of the Penn Treebank (Marcus et al., 1993) and modified versions of the ATIS and Brown corpora (Dahl et al., 1994; Francis and Kucera, 1979). Modification consists in deleting all subjects in the sentences and deleting the determiners. To maintain accuracy on whole sentences, the original version of the modified corpora is added to the training set as well. The adaptation raises the accuracy for whole phrase structure trees to 72%, and the direct object link precision to 90%.

Out of those parses, we can now extract all noun phrases for further processing. The last step for participant identification consists in adding the "implicit protagonist" whenever the subject position in the parse tree is empty.

4.2 Participant Description Sets

The next task consists in the actual learning of script participants, more specifically: We will propose a method that groups participant descriptions occurring in the ESDs for a given scenario into participant description sets (PDSs) that comprise different mentions of one participant.

We assume that two token-identical participant descriptions always have the same word sense and represent the same participant, not only in one ESD, but across all event descriptions within a scenario. This extends the common "one sense per discourse" heuristic (Gale et al., 1992) with a "one participant per sense" assumption on top of that. The resulting loss of precision is only minimal, and we can take participant description types (PTs) rather than tokens to be basic entities, which drastically reduces the size of the basic entity set.

We also exploit structural information given in the alignment tables: If two PTs occur in aligned event descriptions, we take this as a piece of evidence that they belong to the same participant. In the example of Fig. 1, this supports identification of "time" and "desired length".

We complement this structural indicator by semantic similarity information: In the example of Fig. 1, the identification of "bowl" and "dish" is supported by WordNet hyponymy. We use semantic similarity information in different ways:

- WordNet synonymy of PTs, as well as synonymy and direct hyponymy of the head of multiword PTs (like *full can* and *full container*) guarantee participant identity
- A WordNet based semantic similarity score is used as a soft indicator of participant identity

We combine all these information sources by modeling the equivalence-class problem as an Integer Linear Program (Wolsey, 1998, *ILP*). An ILP computes an assignment of integer values to a set of variables, maximizing a given *objective function*. Additional linear equations and inequalities can constrain the possible value assignments.

The problem we want to solve is to determine for each pair pt_i and pt_j in the set of PTs

$\{\mathrm{pt}_1, \ldots, \mathrm{pt}_n\}$ whether they belong to the same equivalence class. We model this in our ILP by introducing variables x_{ij} which can take the values 0 or 1; if x_{ij} takes the value 1 in a solution of the ILP, this means that the tokens of pt_i and the tokens of pt_j belong to the same PDS.

Objective function

We use the objective function to encode semantic similarity and structural information from the alignment. We require the ILP solver to maximize the value of the following linear term:

$$\sum_{i,j=1,i\neq j}^{n} (\mathrm{struc}(pt_i, pt_j) \cdot \mathrm{sim}(pt_i, pt_j) - \theta) \cdot x_{ij} \tag{1}$$

$\mathrm{sim}(i, j)$ stands for the semantic similarity of pt_i and pt_j and is computed as follows:

$$\mathrm{sim}(pt_i, pt_j) = \begin{cases} lin(pt_i, pt_j) + \eta & \text{if } pt_i \text{ and } pt_j \\ & \text{are hyponyms} \\ lin(pt_i, pt_j) & \text{otherwise} \end{cases} \tag{2}$$

For computing similarity, we use Lin's (WordNet-based) similarity measure (Lin, 1998; Fellbaum, 1998), which performs better than several distributional measures which we have tried. Direct hyponymy is a particularly strong indicator; therefore we add the empirically determined constant η to sim in this case.

θ is a cutoff which is also optimized empirically. Every pair with a similarity lower than θ adds a negative score to the objective function when its variable is set to 1. In the final solution, pairs with a similarity score smaller than θ are thus avoided whenever possible.

$\mathrm{struc}(i, j)$ encodes structural information about pt_i and pt_j, i.e. how tokens of pt_i and pt_j are related in the alignment table. Eq. 3 defines this:

$$\mathrm{struc}(pt_i, pt_j) = \begin{cases} \lambda_+ & \text{if } pt_i \text{ and } pt_j \text{ from} \\ & \text{same row} \\ \lambda_- & \text{if } pt_i \text{ and } pt_j \text{ from} \\ & \text{same column and unrelated} \\ 1 & otherwise \end{cases} \tag{3}$$

If pt_i and pt_j are aligned at least once (i.e., their enclosing event descriptions are paraphrase candidates), $\mathrm{struc}(i, j)$ takes a constant value λ_+ greater than 1, thus boosting the similarity of pt_i and pt_j. If the tokens of pt_i and pt_j occur in the same *column* (i.e., they are alternately used by the same subject in an ESD) and the two types have no direct WordNet link, $\mathrm{struc}(pt_i, pt_j)$ takes a constant value smaller than 1 (λ_-) and lowers the similarity score. Both values are empirically optimized.

Hard Constraints

We add a constraint $x_{ij} = 1$ for a pair i, j if one of the following conditions holds:

- pt_i and pt_j share a synset in WordNet
- pt_i and pt_j have the same head (like *laundry machine* and *machine*)
- pt_i and pt_j are both multiword expressions, their modifiers are identical and their heads are either synonyms or hyponyms

Furthermore, if pt_i is the implicit protagonist, we add the constraint $x_{ij} = 1$ if pt_j is a first or second person pronoun, and $x_{ij} = 0$ otherwise.

Finally, we ensure that the ILP groups the participant types into equivalence classes by enforcing symmetry and transitivity. Symmetry is trivially encoded by the following constraint over all i and j:

$$x_{ij} = x_{ji} \tag{4}$$

Transitivity can be guaranteed by adding the following constraints for each i, j, k:

$$x_{ij} + x_{jk} - x_{ik} \leq 1 \tag{5}$$

This is a standard formulation of transitivity, used e.g. by Finkel and Manning (2008).

5 Evaluation

We evaluate our system against a gold standard of 10 scenarios. On average, one scenario consists of 180 event descriptions, containing 54 participant description types realized in 233 tokens. The scenarios are EAT AT A FAST FOOD RESTAURANT, RETURN FOOD (IN A RESTAURANT), PAY WITH CREDIT CARD, TAKE A SHOWER, FEED A PET DOG, MAKE COFFEE, HEAT SOMETHING IN A MICROWAVE, MAIL A LETTER, BUY SOMETHING FROM A VENDING MACHINE, and DO LAUNDRY. The VENDING MACHINE and LAUNDRY scenarios were used for parameter optimization. The parameter values we determined were $\theta = 5.3, \eta = 0.8, \lambda_+ = 3.4$ and $\lambda_- = 0.4$. We solve the ILP using LPSolve (Berkelaar et al., 2004).

SCENARIO	PRECISION				RECALL				F-SCORE			
	full	sem	align	base	full	sem	align	base	full	sem	align	base
LAUNDRY*	0.85	0.76	0.53	0.93	0.75	0.83	0.89	0.57	**0.80**	0.79	0.67	0.70
VENDING M.*	0.80	0.74	0.57	0.84	0.78	0.83	0.97	0.62	**0.79**	0.78	0.72	0.72
FAST FOOD	0.82	0.65	0.55	0.87	0.82	0.85	0.84	0.70	**0.82**	0.74	0.66	0.78
RETURN FOOD	0.80	0.78	0.53	0.88	0.44	0.52	0.63	0.34	0.57	***0.62***	0.57	0.49
COFFEE	0.85	0.77	0.53	0.92	0.80	0.81	0.98	0.68	**0.82**	0.79	0.68	0.78
FEED DOG	0.81	0.67	0.53	0.90	0.88	0.92	0.94	0.57	**0.84**	0.78	0.68	0.70
MICROWAVE	0.89	0.78	0.55	0.93	0.84	0.84	0.89	0.70	**0.86**	0.81	0.68	0.80
CREDIT CARD	0.90	0.82	0.60	0.94	0.54	0.54	0.64	0.40	**0.67**	0.65	0.62	0.56
MAIL LETTER	0.92	0.78	0.54	0.96	0.88	0.88	0.93	0.74	**0.90**	0.83	0.68	0.84
SHOWER	0.87	0.79	0.57	0.94	0.83	0.83	0.86	0.66	**0.85**	0.81	0.69	0.77
AVERAGE*	0.85	0.75	0.55	0.91	0.75	0.79	0.86	0.60	• **0.79**	• 0.76	0.66	0.71
AVERAGE	0.86	0.76	0.55	0.92	0.75	0.77	0.84	0.60	• **0.79**	0.75	0.66	0.71

Figure 2: Results for the full system, the system without structural constraints (sem), the system with structural information only (align) and the naive baseline. Participant descriptions with the right head are considered correct. Starred scenarios have been used for parameter optimization, *average** includes those scenarios, the unmarked average doesn't. A black dot (•) means that the difference to the next lower baseline is significant with $p < 0.05$. The difference between full and base is significant at $p < 0.001$.

5.1 Gold Standard

We preprocessed the 10 evaluation scenarios by aligning them with the RKP algorithm. Two annotators then labeled the 10 aligned scenarios, recording which noun-phrases referred to the same participant. Specifically, the labelers were shown, in order, the sets of aligned event descriptions. For instance, for the microwave script, they would first encounter all available alternative descriptions for putting food on some dish. From each aligned description, the annotators extracted the participant-referring NPs, which were then grouped into blocks of coreferent mentions. After all sets of component-event descriptions had been processed, the annotators also manually sorted the previously extracted blocks into coreferent sets. Implicit participants, typically missing subjects, were annotated, too. For the evaluation, we include missing subjects but do not consider other implicit participants. Each annotator labeled 5 of the scenarios independently, and reviewed the other annotator's work. Difficult cases, mostly related to metonymies, were solved in consultation.

5.2 Baseline and Scoring Method

The system sorts participant descriptions into their equivalence classes, thus we evaluate whether the equivalence statements are correct and whether the classes it found are complete. Speaking in terms of participant description sets, we evaluate the purity of each set (whether all items in a set belong there) and the set completeness (whether another set should have been merged into the current one).

5.2.1 Baselines

We compare our system with three baselines: As a naïve baseline ($base$), we group participant descriptions together only if they are string-equal. This is equivalent to just employing the type-abstraction step we took in the full system and ignoring other information sources.

Additionally, we show the influence of the structural information with a more informed baseline (sem): we replicate our full system but just use the semantic similarity including all hard constraints, without any structural information from the alignment tables. This is equivalent to setting $\mathrm{struc}(i,j)$ in equation 1 always to 1.

In order to show that semantic similarity and the alignment indeed provide contrastive knowledge, we test a third baseline that contains the structural information only ($align$). Here we group all noun phrases i and j together if $\mathrm{struc}(i,j) > 1$ and the pair (i,j) meets all hard constraints.

All parameters for the baselines were optimized separately using the same scenarios as for the full system.

5.2.2 Scoring Method

Because the equivalence classes we compute are similar to coreference sets, we apply the b^3 evaluation metric for coreference resolution (Bagga and Baldwin, 1998). b^3 defines precision and recall as follows: for every token t in the annotation, take the coreference set C_t it is assigned to. Find the

np-matching	PRECISION				RECALL				F-SCORE			
	full	sem	align	base	full	sem	align	base	full	sem	align	base
Gold Tokens	0.92	0.81	0.54	0.97	0.86	0.88	0.96	0.71	0.89	0.84	0.70	0.81
Matching Head	0.86	0.76	0.55	0.92	0.75	0.77	0.84	0.60	0.79	0.75	0.66	0.71
Strict Matching	0.82	0.74	0.52	0.91	0.70	0.71	0.77	0.59	0.74	0.71	0.62	0.71

Figure 3: Averaged evaluation results for three scoring methods: *Gold Tokens* uses gold standard segmentation. *Matching head* uses parsing for PD extraction and phrases with the right head are considered correct. *Strict* requires the whole phrase to match.

set C_{t+gold} that contains t in the gold standard, and assign $precision_t$ and $recall_t$:

$$precision_t = \frac{|C_t \cap C_{t+gold}|}{|C_t|} \quad (6)$$

$$recall_t = \frac{|C_t \cap C_{t+gold}|}{|C_{t+gold}|} \quad (7)$$

Overall precision and recall is averaged over all tokens in the annotation. Overall F_1 score is then computed as follows:

$$F_1 = \frac{2 * precision * recall}{precision + recall} \quad (8)$$

Unlike in coreference resolution, we have the problem that we compare gold-standard annotations against tokens extracted from automatic parses. However, the b^3-metric is only applicable if the gold standard and the test data contain the same set of tokens. Thus we apply b^3_{sys}, a variant of b^3 introduced by Cai and Strube (2010). b^3_{sys} extends the gold standard and the test set such that both contain the same set of tokens. Roughly speaking, every token that appears in the gold standard but not in the test set is copied to the latter and treated as singleton set, and vice versa. See Cai and Strube for details.

With the inaccurate parser, noun phrases are often parsed incompletely, missing modifiers or relative clauses. We therefore consider a participant description as equivalent with a gold standard phrase if they have the same head. This relaxed scoring metric evaluates the system realistically by punishing parsing errors only moderately.

5.3 Results

5.3.1 Scores

Figure 2 shows the results for our system and three baselines. *full* marks the complete system, *sem* is the baseline without structural information, *align* uses exclusively structural information and *base* is the naïve string matching baseline. The starred scenarios were used for parameter optimization and excluded from the final average score. (The AVERAGE* row includes those scenarios.) In terms of the average F-Score, we outperform the baselines significantly ($p < 0.05$, paired two-sample t-test on the f-scores for the different scenarios) in all three cases. The system difference to the naïve baseline even reaches a significance level of $p < 0.001$. While the naïve baseline always gets the best precision results, the $align$-baseline performs best for recall. The latter is due to the numerous alignment errors, which sometimes lead to a simple partition in subjects and objects. Our system finds the best tradeoff between precision and recall, gaining 15% recall on average compared to the naïve baseline and just losing about 6% precision. sem and the naïve baseline differ only moderately. This shows that semantic similarity information alone is not sufficient for distinguishing the different participant descriptions, and that the exploitation of structural information is crucial. However, the structural information by itself is worthless: high precision loss makes $align$ even worse than the naïve baseline.

Fig. 3 compares the same-head scoring metric described in the previous section *(Matching Head)* against two other approaches of dealing with wrongly recognized NP tokens: *Strict Matching* only accepts two NP tokens as equivalent if they are identical; *Gold Tokens* means that our PDS identification algorithm runs directly on the gold standard tokens. This shows that parsing accuracy has a considerable effect on the overall performance of the system. However, our system robustly outperforms the baselines regardless of the matching approach.

5.3.2 Example Output

Fig. 4 illustrates our system's behavior showing its output for the MICROWAVE scenario. Each rectangle on the left represents one PDS, which we rep-

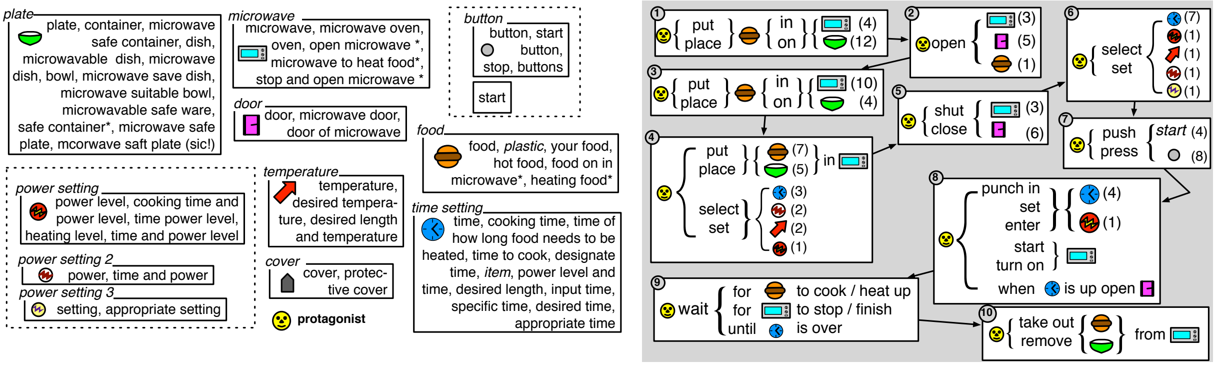
plate
plate, container, microwave safe container, dish, microwavable dish, microwave dish, bowl, microwave save dish, microwave suitable bowl, microwavable safe ware, safe container*, microwave safe plate, mcorwave saft plate (sic!)
microwave
microwave, microwave oven, oven, open microwave *, microwave to heat food*, stop and open microwave *
door
door, microwave door, door of microwave
button
button, start button, stop, buttons
start
food
food, plastic, your food, hot food, food on in microwave*, heating food*
power setting
power level, cooking time and power level, time power level, heating level, time and power level
power setting 2
power, time and power
power setting 3
setting, appropriate setting
temperature
temperature, desired temperature, desired length and temperature
cover
cover, protective cover
protagonist
time setting
time, cooking time, time of how long food needs to be heated, time to cook, designate time, item, power level and time, desired length, input time, specific time, desired time, appropriate time
1
put
place
in
on
(4)
(12)
2
open
(3)
(5)
(1)
3
put
place
in
on
(10)
(4)
4
put
place
(7)
(5)
in
select
set
(3)
(2)
(2)
(1)
5
shut
close
(3)
(6)
6
select
set
(7)
(1)
(1)
(1)
(1)
7
push
press
start (4)
(8)
8
punch in
set
enter
(4)
(1)
start
turn on
when
is up open
9
wait
for
to cook / heat up
for
to stop / finish
until
is over
10
take out
remove
from

Figure 4: The participants we extracted for the MICROWAVE scenario, and a participant-annotated excerpt from the original graph. Descriptions in *italics* indicate sorting mistakes, asterisks (*) mark parsing mistakes. Dotted boxes frame PDSs that actually belong together but were not combined by the algorithm.

resent by an icon in the graph to the right.[2] The participant types in the sets are ordered by frequency, starting with the most frequent one. The labels of the sets are script role labels and were introduced for readability. Note that the structural alignment information allows us to correctly classify *plate* and *container*, and *stop* and *button*, as equivalent, although they are not particularly similar in WordNet. However, especially for rare terms, our algorithm seems too strict: it did not combine the three *power setting* PDSs. Also, we cannot tell start from stop buttons, which is mainly due to the fact that most people did not distinguish them at all but just called them *button(s)* (some microwaves just have one button). The separate grouping of *start* is also related to parsing errors: *start* was mostly parsed as a verb, even when used as object of *push*.

The right part of Fig. 4 shows a version of the RKP temporal script graph for this scenario, with all NP tokens replaced by icons for their PDSs. Ten of its nodes are shown with their temporal ordering, marked by the edges and additionally with encircled numbers. Alternative PDSs are marked with their absolute frequencies. As the subject is always left out in the example data, we assume an implicit protagonist in all cases. The figure demonstrates that we can distinguish the participants, even though the event alignment has errors.

6 Conclusion

We have presented a system that identifies script participants from unlabeled data by grouping equivalent noun phrases together. Our system combines semantic similarity and global structural information about event alignments in an ILP. We have shown that the system outperforms baselines that are restricted to each of these information sources alone; that is, both structural and similarity information are essential.

We believe that we can improve our system in a number of ways, e.g. by training a better parser or switching to a more sophisticated semantic similarity measure. One particularly interesting direction for future work is exploiting participant information to improve the alignments; this would allow us to merge the "put food in microwave" nodes in the graph of Fig. 4, which look identical once noun phrases have been abstracted into participants. We could achieve this by jointly modeling the event alignment problem and the participant identification problem in the same ILP.

While our approach to learning participants is unsupervised once some parameters have been optimized on a small amount of labeled data, we can only obtain a large-scale knowledge base of scripts if we can collect large amounts of scenario descriptions. Thus the next step must demonstrate that this can be done, without requiring the manual selection of scenarios to ask people about. A promising approach is collecting data through online games; this has been shown to be successful in other domains (e.g. by von Ahn and Dabbish (2008)), and we are optimistic that we can apply this here as well.

Acknowledgments

We're indebted to Ines Rehbein who carryied out the parser experiments. We thank the anonymous reviewers and particularly Nate Chambers for their

[2] We omit some PDSs in the presentation for lack of space.

helpful comments. – This work was funded by the Cluster of Excellence “Multimodal Computing and Interaction” in the German Excellence Initiative, and the SALSA project of the German Research Foundation DFG.

References

Ernst Althaus, Nikiforos Karamanis, and Alexander Koller. 2004. Computing locally coherent discourses. In *Proc. of ACL-04*.

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *Proc. of LREC-98*.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The berkeley framenet project. In *Proc. of COLING-98*.

Avron Barr and Edward Feigenbaum. 1981. *The Handbook of Artificial Intelligence, Volume 1*. William Kaufman Inc., Los Altos, CA.

Regina Barzilay and Mirella Lapata. 2006. Aggregation via set partitioning for natural language generation. In *Proc. of HLT-NAACL 2006*.

Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2010. Global learning of focused entailment graphs. In *Proc. of ACL-10*.

Michel Berkelaar, Kjell Eikland, and Peter Notebaert. 2004. lp_solve, a Mixed Integer Linear Programming (MILP) solver Version 5.0. Website.

Jie Cai and Michael Strube. 2010. Evaluation metrics for end-to-end coreference resolution systems. In *Proc. of SIGDIAL 2010*.

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proc. of ACL-08: HLT*.

Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised learning of narrative schemas and their participants. In *Proc. of ACL-IJCNLP 2009*.

Richard Edward Cullingford. 1977. *Script application: computer understanding of newspaper stories*. Ph.D. thesis, Yale University, New Haven, CT, USA.

Deborah A. Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg. 1994. Expanding the scope of the ATIS task: the ATIS-3 corpus. In *Proceedings of the HLT-94*, HLT ’94.

Pascal Denis and Jason Baldridge. 2007. Joint determination of anaphoricity and coreference resolution using integer programming. In *Proc. of HLT-NAACL 2007*.

Richard Durbin, Sean Eddy, Anders Krogh, and Graeme Mitchison. 1998. *Biological Sequence Analysis*. Cambridge University Press.

Christiane Fellbaum. 1998. *WordNet: An Electronical Lexical Database*. The MIT Press, Cambridge, MA.

Jenny Rose Finkel and Christopher D. Manning. 2008. Enforcing transitivity in coreference resolution. In *Proc. of ACL-08: HLT*.

W. N. Francis and H. Kucera, 1979. *Manual of Information to Accompany a Standard Corpus of Present-Day Edited American English, for use with Digital Computers*. Department of Linguistic, Brown University.

William A. Gale, Kenneth W. Church, and David Yarowsky. 1992. One sense per discourse. In *Proceedings of the workshop on Speech and Natural Language*.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proc. of ACL-03*.

Dekang Lin. 1998. An information-theoretic definition of similarity. In *Proc. of ICML-98*.

Inderjeet Mani, Marc Verhagen, Ben Wellner, Chong Min Lee, and James Pustejovsky. 2006. Machine learning of temporal relations. In *Proc. of COLING/ACL-2006*.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: the penn treebank. *Comput. Linguist.*, 19.

Erik T. Mueller. 2004. Understanding script-based stories using commonsense reasoning. *Cognitive Systems Research*.

James Pustejovsky, Marc Verhagen, Roser Sauri, Jessica Littman, Robert Gaizauskas, Graham Katz, Inderjeet Mani, Robert Knippen, and Andrea Setzer. 2006. TimeBank 1.2. Linguistic Data Consortium.

Lisa F. Rau, Paul S. Jacobs, and Uri Zernik. 1989. Information extraction and text summarization using linguistic knowledge acquisition. *Information Processing and Management*.

Michaela Regneri, Alexander Koller, and Manfred Pinkal. 2010. Learning script knowledge with web experiments. In *Proc. of ACL-10*.

Roger C. Schank and Robert P. Abelson. 1977. *Scripts, Plans, Goals and Understanding*. Lawrence Erlbaum, Hillsdale, NJ.

Reid Swanson and Andrew S. Gordon. 2008. Say anything: A massively collaborative open domain story writing companion. In *Proc. of ICIDS 2008*.

Luis von Ahn and Laura Dabbish. 2008. Designing games with a purpose. *Commun. ACM*, 51(8).

Laurence Wolsey. 1998. *Integer programming*. Wiley-Interscience.