Aligning Script Events with Narrative Texts

Simon Ostermann[†] **Michael Roth**[†‡] **Stefan Thater**[†] **Manfred Pinkal**[†]
[†] Saarland University [‡] University of Illinois at Urbana-Champaign
{simono|mroth|stth|pinkal}@coli.uni-saarland.de

Abstract

Script knowledge plays a central role in text understanding and is relevant for a variety of downstream tasks. In this paper, we consider two recent datasets which provide a rich and general representation of script events in terms of paraphrase sets. We introduce the task of mapping event mentions in narrative texts to such script event types, and present a model for this task that exploits rich linguistic representations as well as information on temporal ordering. The results of our experiments demonstrate that this complex task is indeed feasible.

1 Introduction

Event structure is a prominent topic in NLP. While semantic role labelers (Gildea and Jurafsky, 2002; Palmer et al., 2010) are well-established tools for the analysis of the internal structure of event descriptions, modeling relations between events has gained increasing attention in recent years. Research on event coreference (Bejan and Harabagiu, 2010; Lee et al., 2012), temporal event ordering in newswire texts (Ling and Weld, 2010), as well as shared tasks on cross-document event ordering (Minard et al., 2015, inter alia) have in common that they model cross-document relations.

The focus of this paper is on the task of analyzing text-internal event structure. We share the view of a long tradition in NLP (see e.g. Schank and Abelson (1975); Chambers and Jurafsky (2009); Regneri et al. (2010)) that *script knowledge* is of central importance to this task, i.e. common-sense knowledge about events and their typical order in everyday activities (also referred to as *scenarios*, Barr and Feigenbaum (1981)). Script knowledge guides expectation by predicting which type of event or discourse referent might be addressed next in a story (Modi et al., 2017), allows to infer missing events from events explicitly mentioned (Chambers and Jurafsky, 2009; Jans et al., 2012; Rudinger et al., 2015), and to determine text-internal temporal order (Modi and Titov, 2014; Frermann et al., 2014).

We address the task of automatically mapping narrative texts to scripts, which will leverage explicit script knowledge for the afore-mentioned aspects of text understanding, as well as for downstream tasks such as textual entailment, question answering or paraphrase detection. We build on the work of Regneri et al. (2010) and Wanzare et al. (2016), who collect explicit script knowledge via crowdsourcing, by asking people to describe everyday activities. These crowdsourced descriptions form a basis for high-quality automatic extraction of script structure without any human intervention (Regneri et al., 2010; Wanzare et al., 2017). The events of the resulting structure are defined as sets of alternative realizations, which cover lexical variation and provide paraphrase information. To the best of our knowledge, these advantages have not been explicitly used elsewhere.

Aligning script structures with texts is a complex task. In a first attempt, we assume that three steps are necessary to solve it, although in the long run, an integrated approach will be preferable: First, the script which is addressed by the event mention must be identified. Second, it has to be decided whether a verb denotes a script event at all. Finally, event verbs need to be assigned a script-specific event type label. This work focuses on the last two steps: We use a corpus of narrative stories each of which is centered around a specific script scenario, and distinguish verbs related to the central script from all other verb occurrences with a simple decision tree classifier. We then train a sequence labeling model only on crowdsourced script data and assign event type labels to all script-related event verbs.

Our results substantially outperform informed

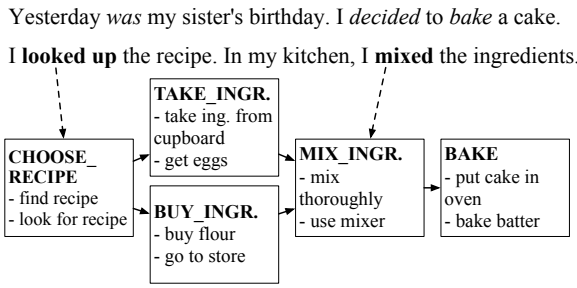

Figure 1: An example of text-to-script mapping with an excerpt of the BAKING A CAKE script and a story snippet.

baselines, in spite of the availability of only small amounts of training data. In particular, we also demonstrate the relevance of event ordering information provided by script knowledge.

Our code and all data and parameters that are used are publicly available under https://github.com/SimonOst.

2 Task and Data

As a basis for the task of text-to-script mapping, we make use of two recently published datasets. *DeScript* (Wanzare et al., 2016) is a collection of crowdsourced linguistic descriptions of event patterns for everyday activities, so called *event sequence descriptions (ESDs)*. ESDs consist of short telegram-style descriptions of single events (*event descriptions, ED*). The textual order of EDs corresponds to the temporal order of respective events, i.e. temporal information is explicitly encoded. *DeScript* contains 50 ESDs for each of 40 different scenarios. Alongside the ESDs, it also provides gold event paraphrase sets, i.e. clusters of all event descriptions denoting the same event type, labeled with the respective type.

While DeScript is a source of structured script knowledge, the *InScript* corpus (Modi et al., 2016) provides us with the appropriate kind of narrative texts. *InScript* is a collection of 910 stories centered around some specific scenario, for 10 of the 40 scenarios in *DeScript*, e.g. BAKING A CAKE, RIDING A BUS, TAKING A SHOWER. All verbs occurring in the texts are annotated with an event type if they are relevant to the script instantiated by the story; as *non-script event* otherwise.

In the upper part of Fig. 1, you see the initial fragment of a story about baking a cake; together with a script excerpt in the lower part, depicted by labeled event paraphrase sets. *I looked up the recipe* and *I mixed the ingredients* mention relevant script events, and therefore should be labeled with the indicated event types (CHOOSE_RECIPE, MIX_INGREDIENTS). Fig. 1 also illustrates the potential of text-to-script mapping: script knowledge enables to predict that a baking event might be addressed next in the story. The verb *was* does not denote an event at all, and *decide* is not part of the BAKING A CAKE script, so they are assigned the label *non-script event*. Actually, *InScript* comes with two additional categories of verbs (*script-related* and *script-evoking*), which we subsume under *non-script event*.

The central task addressed in our paper, the automatic labeling of all script-relevant verbs in the *InScript* text with a script-specific event type, uses only *DeScript* data for training; event-type labels of *InScript* are used for evaluation purposes only.

3 Model

Section 3.1 defines the central part of our system, a sequence model for classifying script-relevant verbs into scenario-specific event types. For full automation of the text-to-script mapping, we describe in Section 3.2 a model for identifying script-relevant verbs.

3.1 Event Type Classification

For identifying the correct event type given a script-relevant verb, we leverage two types of information: We require a representation for the meaning and content of the event mention, which takes into account not only the verb, but also the persons and objects involved in an event, i.e. the *script participants*. In addition, we take event ordering information into account, which helps to disambiguate event mentions based on their local context. To model both event types and sequences thereof, we implement a linear-chain conditional random field (CRF, Lafferty et al. (2001)). Our implementation is based on the CRF++ toolkit[1] and employs two types of features:

Sequential Feature. Our CRF model utilizes event ordering information in the form of binary indicator features that encode the co-occurrence of two event type labels in sequence.

Meaning Representation Features. Two feature types encode the meaning of a textual event mention. One is a shallow form of representation derived from precomputed word embeddings

[1] taku910.github.io/crfpp/

(*word2vec*, Mikolov et al. (2013)). This feature type captures distributional information of the verb and its direct nominal dependents[2], which we assume to denote script participants, and is computed by averaging over the respective word vector representations.[3] We use pretrained 300-dimensional embeddings that are trained on the Google News corpus.[4]As a more explicit but sparse form of content representation, we use as the other type of feature the lemma of the verb, its indirect object and its direct object.

3.2 Identifying Script-Relevant Verbs

We use a decision tree classifier for identifying script-relevant verbs (*J48* from the Weka toolkit, Frank et al. (2016)) that takes into account four classes: the three *non-script event* classes from *InScript* and one class for all *event-verbs*. At test time, the three *non-script event* classes are merged into one class. Due to the lack of *non-script event* instances in *DeScript*, we train and test our model on all verbs occurring in *InScript*. We use the following feature types:

Syntactic Features. We employ syntactic features for identifying verbs that only rarely denote script events, independent of the scenario: a feature for auxiliaries; for verbs that govern an adverbial phrase (mostly if-clauses); a feature indicating the number of direct and indirect objects; and a lexical feature that checks if the verb belongs to a predefined list of non-action verbs.

Script Features. For finding verbs that match the current script scenario, we employ two features: a binary feature indicating whether the verb is used in the ESDs for the given scenario; and a scenario-specific tf–idf score that is computed by treating all ESDs from a scenario as one document, summed over the verb and its dependents. In Section 4.2, we evaluate models with and without script features, to test the impact of scenario-specific information.

Frame Feature. We further employ frame-semantic information because we expect script events to typically evoke certain frames.We use a state-of-the-art semantic role labeler (Roth, 2016; Roth and Lapata, 2016) based on *FrameNet* (Ruppenhofer et al., 2006) to predict frames for all verbs, encoding the frame as a feature. We address sparsity of too specific frames by mapping all frames to higher-level super frames using the *framenet querying package*[5].

[2]For EDs, we use all mentioned head nouns.

[3]To emphasize the importance of the verb, we double its weight when averaging.

[4]Because our CRF model only supports nominal features, we discretize embeddings from code.google.com/archive/p/word2vec/ by binning the component values into three intervals $[-\infty, -\epsilon]$, $[-\epsilon, \epsilon]$, $[\epsilon, \infty]$. The hyperparameter ϵ is determined on a held-out development set.

	P	R	F_1
Lemma	0.365	**0.949**	0.526
Our model	**0.628**	0.817	**0.709**
Our model (scen. indep.)	0.513	0.877	0.645

Table 1: Identification of script-relevant verbs within a scenario and independent of the scenario.

4 Evaluation

4.1 Experimental Setup

We evaluate our model for text-to-script mapping based on the resources introduced in Section 2. We process the *InScript* and *DeScript* data sets using the Stanford Parser (Klein and Manning, 2003)[6]. We further resolve pronouns in *InScript* using annotated coreference chains from the gold standard.

We individually test the two components, i.e. the identification of script-relevant verbs and event classification. Experiments on the first sub-task are described in Section 4.2. Sections 4.3 and 4.4 present results on the latter task and a combination of both tasks, respectively.

4.2 Identifying Script-Relevant Verbs

In this evaluation, we test the ability of our model to identify verbs in narrative texts that instantiate script events. Our experiments make use of a 10-fold cross-validation setting within all texts of one scenario. To test the model in a scenario-independent setting, we perform additional experiments based on a cross-validation with the 10 scenarios as one fold each and exclude the script features. That is, we repeatedly train our model on 9 scenarios and evaluate on the remaining scenario, without using any information about the test scenario.

Models. We compare the model described in Section 3.2 to a baseline (*Lemma*) that always assigns the *event* class if the verb lemma is mentioned in *DeScript*. We report precision, recall and F_1-score on event verbs, averaged over all scenarios.

[5]github.com/icsi-berkeley/framenet

[6]To improve performance on the simplistic sentences from *DeScript*, we follow Regneri (2013) and re-train the parser.

Results. Table 1 gives an overview of the results based on 10-fold cross-validation. Our scenario-specific model is capable of identifying more than 81% of script-relevant verbs at a precision of about 63%. This is a notable improvement over the baseline, which identifies 94.9% of the event verbs, but at a precision of only 36.5%.

The table also gives numbers for the scenario-independent setting: Precision drops to around 51% if only training data from other scenarios is available. One of the main difficulties here lies in classifying different *non-script event* verb classes in a way that generalizes across scenarios. Modi et al. (2016) also found that distinguishing specific types of non-script events from script events can be difficult even for humans.

4.3 Event Type Classification

In this section, we describe experiments on the text-to-script mapping task based on the subset of event instances from *InScript* that are annotated as script-related. As training data, we use the *ESDs* and the event type annotations from the *DeScript* gold standard[7]. The evaluation task is to classify individual event mentions in *InScript* based on their verbal realization in the narrative text. We evaluate against the gold-standard annotations from *InScript*. Since event type annotations are used for evaluation purposes only, this task comes close to a realistic setup, in which script knowledge is available for specific scenarios but no training data in the form of event-type annotated narrative texts exists.

Models. We evaluate our CRF model described in Section 3.1 against two baselines that are based on textual similarity. Both baselines compare the event verb and its dependents in *InScript* to all EDs in *DeScript* and assign the event type with the highest similarity. *Lemma* is a simple measure based on word overlap, *word2vec* uses the same embedding representation as the CRF model (before discretization) but simply assigns the best matching event type label based on cosine similarity. We report precision, recall and F_1-scores, macro-averaged over all script-event types and scenarios.

Results. Results for all models are presented in Table 2. Our CRF model achieves a F_1-score of 0.545, a considerably higher performance in comparison to the baselines. As can be seen from excluding the sequential feature, ordering information improves the result. The rather small difference is due to the fact that ordering information can also be misleading (cf. Section 5). We found, however, that including the sequential feature accounts for an improvement of up to 4% in F_1 score, depending on the scenario.

[7] In *DeScript*, there are some rare cases of *EDs* that do not describe a script event, but that are labeled as *non-script event*. We exclude these from the training data.

	P	**R**	**F_1**
Lemma	0.343	0.416	0.374
Word2vec	0.356	0.448	0.395
CRF model	**0.608**	**0.496**	**0.545**
CRF, no seq.	0.599	0.487	0.536

Table 2: Event Type Classification performance, with and without sequential features.

	P	**R**	**F_1**
Ident. model+Lemma	0.253	0.451	0.323
Ident. model+Word2vec	0.255	0.477	0.331
Ident. model+CRF model	**0.445**	**0.520**	**0.479**

Table 3: Full text-to-script mapping results.

4.4 Full Text-to-Script Mapping Task

We now address the full text-to-script mapping task, a combination of the identification of relevant verbs and event type classification. This setup allows us to assess whether the general task of a fully automatic mapping of verbs in narrative texts to script events is feasible.

Models. We compare the same models as in Section 4.3, but use them on top of our model for identifying script-relevant verbs (cf. Section 4.2) instead of using the gold standard for identification.

Results. On the full text-to-script mapping task, our combined identification and CRF model achieves a precision and recall of 0.445 and 0.52, resp. (cf. Table 3). This reflects an absolute improvement over the baselines of 0.148 and 0.156 in terms of F_1-score. The results reflect the general difficulty of this task but are promising overall. As reported by Modi et al. (2016), even human annotators only achieve an agreement of 0.64 in terms of Fleiss' Kappa (1971).

5 Discussion

In this section, we discuss cases in which our system predicted the wrong event type and give examples for each case. We identified three major error sources:

Lexical Coverage. We found that although *DeScript* is a small resource, training a model purely on *ESDs* works reasonably well. Coverage problems can be seen in cases of events for which only few *EDs* exist. An example is the CHOOSE_TREE event (the event of picking a tree at the shop) in the PLANTING A TREE scenario. There are only 3 *EDs* describing the event, each of which uses the event verb "choose". In contrast, we find that "choose" is used in less than 10% of the event mentions in *InScript*. Because of this mismatch, which can be attributed to the small training data size, more frequently used verbs for this event in *InScript*, such as "pick" and "decide", are labeled incorrectly.

We observe that our meaning representation might be insufficient for finding synonyms for about 30% of observed verb tokens. This specifically includes scenario-specific and uncommon verbs, such as "squirt" in the context of the BAKING A CAKE scenario (*squirt the frosting onto the cake*). Problems may also arise from the fact that about 23% of the verb types occur in multiple paraphrase clusters of a scenario.

Misleading Ordering Information. We found that ordering information is in general beneficial for text-to-script alignment. We however also identified cases for which it can be misleading, by comparing the output of our full model to the model that does not use sequential features. As another result of the small size of *DeScript*, there are plausible event sequences that appear only rarely or never in the training data. This error source is involved in 60–70% of the observed misclassifications due to misleading ordering information. An example is the WASH event in the GETTING A HAIRCUT scenario: It never appears directly after the MOVE_IN_SALON event (i.e. walking from the counter to the chair) in *DeScript*, but its a plausible sequence that is misclassified by our model.

In almost 15% of the observed errors, an event type is mentioned more than once, leading to misclassifications whenever ordering information is used. One reason for this might be that events in *InScript* are described in a more exhaustive or fine-grained way. For example, the WASH event in the TAKING A BATH scenario is often broken up into three mentions: wetting the hair, applying shampoo, and washing it again. However, because there is only one event type for the three mentions, this sequence is never observed in *DeScript*.

Events with an interchangeable natural order lead to errors in a number of cases: In the BAKING A CAKE scenario, a few misclassifications happen because the order in which e.g. ingredients are prepared, the pan is greased and the oven is preheated is very flexible, but the model overfits to what it observed from the training.

As last, there are also a few cases in which an event is mentioned, even before it actually takes place. In the case of the *borrowing a book* scenario, there are cases in *InScript* that mention in the first sentence that the purpose of the visit is to return a book. In *DeScript* in contrast, the RETURN event always takes place in the very end.

Near Misses. For many verbs, it is also difficult for humans to come up with one correct event label. By investigating confusion matrices for single scenarios, we found that for at least 3–5% of script event verbs in the test set, our model predicted an "incorrect" label for such verbs, but that label might still be plausible. In the BAKING A CAKE scenario, for example, there is little to no difference between mentions of making the dough and preparing ingredients. As a consequence, these two events are often confused: Approximately 50% of the instances labeled as PREPARE_INGREDIENTS are actually instances of MAKE_DOUGH.

6 Summary

In this paper, we addressed the task of automatically mapping event denoting expressions in narrative texts to script events, based on an explicit script representation that is learned from crowdsourced data rather than from text collections. Our models outperform two similarity-based baselines by leveraging rich event representations and ordering information. We showed that models of script knowledge can be successfully trained on crowdsourced data, even if the number of training examples is small. This work thus builds a basis for utilizing the advantages of crowdsourced script representations for downstream tasks and future work, e.g. paraphrase identification in discourse context or event prediction on narrative texts.

Acknowledgments

We thank the anonymous reviewers for their helpful comments. This research was funded by the German Research Foundation (DFG) as part of SFB 1102 'Information Density and Linguistic Encoding'. Work by MR in Illinois was supported by a DFG Research Fellowship (RO 4848/1-1).

References

Avron Barr and Edward A. Feigenbaum. 1981. *The Handbook of Artificial Intelligence*. Addison-Wesley.

Cosmin Adrian Bejan and Sanda Harabagiu. 2010. Unsupervised event coreference resolution with rich linguistic features. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 1412–1422.

Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised learning of narrative schemas and their participants. *Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP* .

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin* 76(5):378.

Eibe Frank, Mark A. Hall, and Ian H. Witten. 2016. The weka workbench. online appendix for "data mining: Practical machine learning tools and techniques" .

Lea Frermann, Ivan Titov, and Manfred Pinkal. 2014. A hierarchical bayesian model for unsupervised induction of script knowledge. In *EACL*. volume 14, pages 49–57.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational linguistics* 28(3):245–288.

Bram Jans, Steven Bethard, Ivan Vulić, and Marie Francine Moens. 2012. Skip n-grams and ranking functions for predicting script events. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 336–344.

Dan Klein and Christopher D. Manning. 2003. Fast exact inference with a factored model for natural language parsing. In *In Advances in Neural Information Processing Systems 15 (NIPS*. MIT Press, pages 3–10.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*. ICML '01, pages 282–289.

Heeyoung Lee, Marta Recasens, Angel Chang, Mihai Surdeanu, and Dan Jurafsky. 2012. Joint entity and event coreference resolution across documents. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, pages 489–500.

Xiao Ling and Daniel S Weld. 2010. Temporal information extraction. In *AAAI*. volume 10, pages 1385–1390.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR* .

Anne-Lyse Minard, Manuela Speranza, Eneko Agirre, Itziar Aldabe, Marieke van Erp, Bernardo Magnini, German Rigau, Ruben Urizar, and Fondazione Bruno Kessler. 2015. Semeval-2015 task 4: Timeline: Cross-document event ordering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*. pages 778–786.

Ashutosh Modi, Tatjana Anikina, Simon Ostermann, and Manfred Pinkal. 2016. Inscript: Narrative texts annotated with script information. *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 16)* .

Ashutosh Modi and Ivan Titov. 2014. Inducing neural models of script knowledge. In *Proceedings of the Conference on Computational Natural Language Learning (CoNLL)*. Baltimore, MD, USA.

Ashutosh Modi, Ivan Titov, Vera Demberg, Asad Sayeed, and Manfred Pinkal. 2017. Modelling semantic expectation: Using script knowledge for referent prediction. *Transactions of the Association for Computational Linguistics* 5:31–44.

Martha Palmer, Daniel Gildea, and Nianwen Xue. 2010. Semantic role labeling. *Synthesis Lectures on Human Language Technologies* 3(1):1–103.

Michaela Regneri. 2013. *Event Structures in Knowledge, Pictures and Text*. Ph.D. thesis, Universität des Saarlandes.

Michaela Regneri, Alexander Koller, and Manfred Pinkal. 2010. Learning script knowledge with web experiments. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, ACL '10, pages 979–988.

Michael Roth. 2016. Improving frame semantic parsing via dependency path embeddings. In *Book of Abstracts of the 9th International Conference on Construction Grammar*. Juiz de Fora, Brazil, pages 165–167.

Michael Roth and Mirella Lapata. 2016. Neural semantic role labeling with dependency path embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany, pages 1192–1202.

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.

Josef Ruppenhofer, Michael Ellsworth, Miriam RL Petruck, Christopher R Johnson, and Jan Scheffczyk. 2006. Framenet ii: Extended theory and practice.

Roger C Schank and Robert P Abelson. 1975. *Scripts, plans, and knowledge*. Yale University New Haven, CT.

Lilian D. A. Wanzare, Alessandra Zarcone, Stefan Thater, and Manfred Pinkal. 2016. A crowdsourced database of event sequence descriptions for the acquisition of high-quality script knowledge. *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)* .

Lilian D. A. Wanzare, Alessandra Zarcone, Stefan Thater, and Manfred Pinkal. 2017. Inducing script structure from crowdsourced event descriptions via semi-supervised clustering. *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics* .