Mapping Text to Scripts: An Entailment Study

Simon Ostermann, Hannah Seitz, Stefan Thater, Manfred Pinkal
Saarland University
Saarbrücken, Germany
{simono|hseitz|stth|pinkal}@coli.uni-saarland.de

Abstract

Commonsense knowledge as provided by scripts is crucially relevant for text understanding systems, providing a basis for commonsense inference. This paper considers a relevant subtask of script-based text understanding, the task of mapping event mentions in a text to script events. We focus on script representations where events are associated with paraphrase sets, i.e. sets of crowdsourced event descriptions. We provide a detailed annotation of event mention/description pairs with textual entailment types. We demonstrate that representing events in terms of paraphrase sets can massively improve the performance of text-to-script mapping systems. However, for a residual substantial fraction of cases, deeper inference is still required.

Keywords: script knowledge, annotation study, textual entailment

1. Introduction and Motivation

Scripts represent knowledge about everyday activities, or *scenarios*, like "going to the movies" or "having dinner at a restaurant" (Schank and Abelson, 1975). They consist of *events* like ORDER MENU or EAT that take place in the scenario, plus information about the typical temporal order in which these events happen, as well as knowledge about *participants* that play a role in the script, such as the waiter or plates and cutlery. Figure 1 shows a graphical representation of the script of the BAKING A CAKE scenario. The nodes in the graph represent events and the edges their temporal order. Relevant script participants, such as CAKE, KITCHEN or INGREDIENTS, are listed in the lower right corner. The dashed boxes represent possible linguistic realizations of the individual events in terms of *paraphrase sets*, described in more detail below.

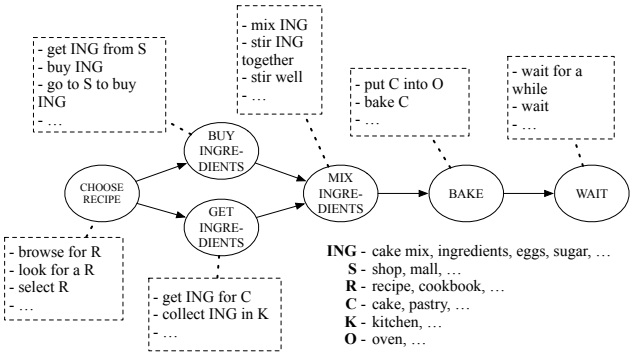

Figure 1: An example for a temporal script graph (BAKING A CAKE).

In communication, script knowledge is assumed to be part of the common ground, and people often do not mention events which can easily be inferred to have happened by the addressee. For instance, if someone tells about the last time they baked a cake, it is likely that they do not mention the fact that the cake was put into the oven, because it is obvious that this event took place. In contrast, a text understanding system that does not have access to script knowledge will probably not be able to draw this inference.

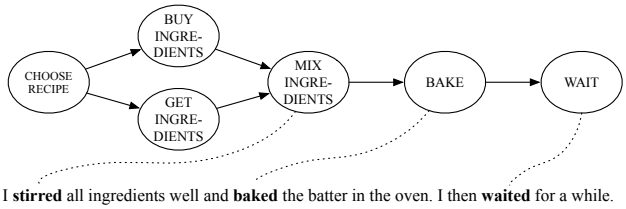

Figure 2: An example for text-to-script mapping in the BAKING A CAKE scenario

Script knowledge has been shown to be useful for a variety of tasks that are required for text understanding, such as event prediction (Chambers and Jurafsky, 2009; Pichotta and Mooney, 2014; Modi et al., 2016), event ordering (Modi and Titov, 2014), paraphrasing (Wanzare et al., 2017) or discourse referent prediction (Modi et al., 2017).

One important aspect in which models that make use of script knowledge differ is the representation of events. Chambers and Jurafsky (2009), for instance, use a "shallow" surface-oriented representation of events which is based on the verb of an event-denoting clause. A different approach has been proposed by Regneri et al. (2010) (henceforth, RKP), who adopt a richer representation of events in terms of *paraphrase sets*, as depicted in Figure 1. Script knowledge of this kind is acquired by first crowdsourcing alternative descriptions of an activity type in terms of sequences of short, telegram-style natural-language *event descriptions* (ED). Then, paraphrase sets are induced automatically as clusters of EDs, using multiple sequence alignment (Regneri et al., 2010) or semi-supervised clustering (Wanzare et al., 2016).

In order to tap the potential of script knowledge in text understanding, systems must be able to link event mentions in texts to the corresponding event types of a script, as indicated by the dotted lines in Figure 2. To our knowledge, Ostermann et al. (2017) is the only existing work on this *text-to-script mapping* task. Their approach is based on RKP-style script representations. Using this representation, the identification of the correct event type of an event mention is in many cases reduced to a simple identity check

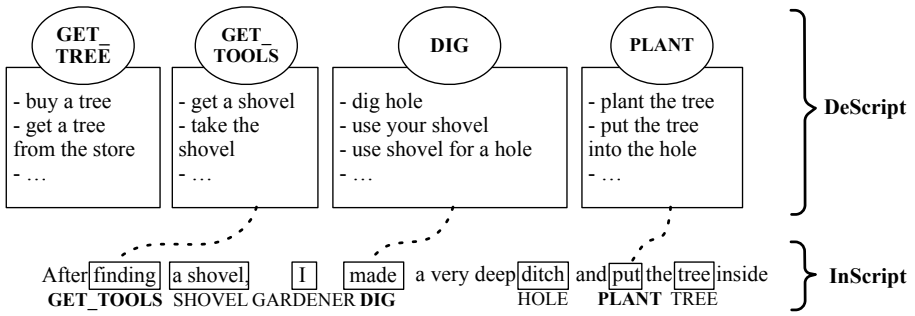

Figure 3: An example annotation in InScript (lower part), and the corresponding paraphrase sets from DeScript (upper part), for the PLANTING A TREE scenario.

between the event mention and one of the EDs in the corresponding paraphrase set. The model achieves promising results when compared to a non-trivial baseline (0.55 F_1 over a baseline of 0.40 F_1), but one would expect even better performance given that the crowdsourced paraphrase sets should substantially facilitate the mapping task.

In this paper, we investigate which types of knowledge and inference are required to assign correct event types to event mentions in text, and to which degree crowdsourced script representations help to facilitate text-to-script mapping. We do so by determining the type of semantic relations between pairs of an event mention and the paraphrase set representing the corresponding event type. In Figure 2, for example, we determine the type of semantic relation that needs to be modeled in order to align the phrase *stir all ingredients* with the MIX_INGREDIENTS paraphrase set.

The contributions of this work are as follows:

- We provide a manual annotation of word-level entailment types on verbs/events and nouns/participants (e.g. *Synonymy*, *Hypernymy*, *Inference* etc., Section 3.) based on an existing corpus of narrative texts and an existing collection of script data. In a second step, we compositionally derive clause-level entailment types (e.g. *Equality*, *Entailment*, etc., Section 4.) that illustrate the types of inference that need to be modeled for the alignment of the clause with an event paraphrase set.
- We provide a detailed analysis of our annotation and show that (1) paraphrase sets as a constitutive part of script representations massively reduce the difficulty of automatic text-to-script mapping and increase its accuracy. We also find that (2) a substantial sub-class of cases cannot be handled using just identity checks or shallow semantic modeling, but require deeper inference methods (Section 5.).
- We demonstrate the usefulness of our dataset as a testbed and diagnostic tool for the differentiated assessment of text-to-script mapping systems, by applying it to the system of Ostermann et al. (2017) (Section 6.).

2. Data

For our study, we use two existing resources that form the basis for evaluating text-to-script mapping: *InScript* (Modi et al., 2016), a collection of narrative texts centered around 10 script scenarios, and *DeScript* (Wanzare et al., 2016), a resource of structured script knowledge in the form of paraphrase sets, covering the same 10 scenarios.

InScript contains 910 stories in total. Verbs and nouns in InScript are manually annotated with participant and event type labels, respectively (see the lower part of Figure 3). The labels are based on manually created, scenario-specific templates, which list all central event and participant types, such as GET_TOOLS or DIG (events) and SHOVEL or GARDENER (participants) for the PLANTING A TREE scenario.

For target script representations, we use DeScript. It contains crowdsourced event sequence descriptions for 40 different everyday scenarios, including the 10 InScript scenarios, as well as manually created paraphrase sets for the latter, which are labeled with the same event types as used in InScript, as can be seen in the upper part of Figure 3: Each event that is labeled in InScript has a corresponding paraphrase set in DeScript. This provides a gold standard alignment between textual event mentions and paraphrase sets, as indicated by the dotted lines, which is the basis for our annotation. The paraphrase sets in this kind of representation contain not only lexical synonyms, but also scenario-specific paraphrases of the same event: *use your shovel* and *dig hole* are not synonyms in a narrow sense, but in the context of PLANTING A TREE, they both describe the DIG event.

While DeScript provides manually created gold event paraphrase sets, the corresponding information on participant level is missing. Since we need entailment relations on the noun phrase/participant level for our compositional derivation of clause-level entailment (see Section 4.), we extended DeScript with paraphrase sets for participant descriptions (as depicted in the lower right corner of Figure 1). For this purpose, we annotated all nouns in the 10 scenario subset of DeScript with labels from the InScript inventory of participant types. Data from the BUS and TREE scenarios were annotated by two annotators to assess the inter-annotator agreement, which was *almost perfect* (Landis and Koch (1977), $\kappa = 0.91$).

From this annotation, paraphrase sets for participants were derived, i.e. the sets of all nouns describing the same participant. In the BAKING A CAKE scenario for example, all different ingredients such as eggs, flour, milk etc. are members of the INGREDIENT paraphrase set.

For our study, we selected 3 out of the 10 scenarios that differ with respect to their complexity: TAKING THE BUS, BAKING A CAKE and PLANTING A TREE. We annotated

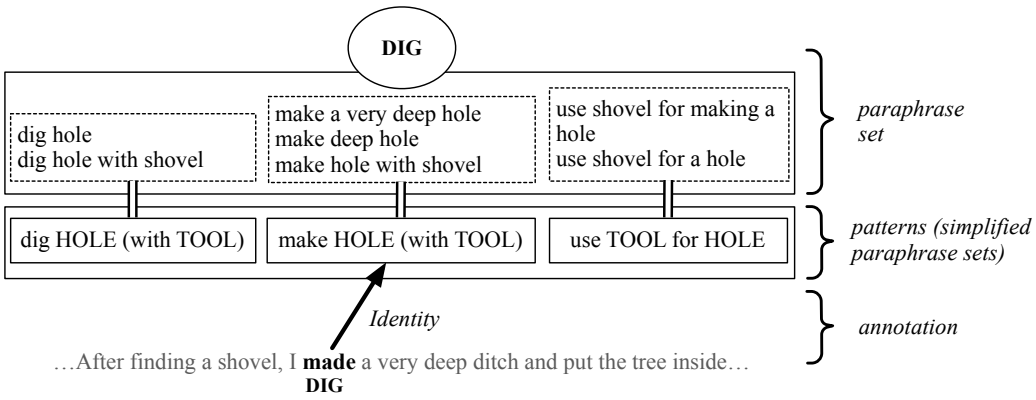

Figure 4: Combining the EDs in the DIG event into patterns (upper part) and the actual verb annotation (lower part).

all script-relevant verb instances and all relevant participant instances in every story.

3. Lexical Entailment: Annotation Study

To identify the type of semantic relation that needs to be modeled in order to align an event-denoting clause in the text with a paraphrase set representing the same event, the most straightforward way would be to conduct a clause-level entailment annotation between the clause and EDs in the paraphrase set, e.g. with a set of clausal entailment types similar to the ones used in (MacCartney and Manning, 2009).

We found, however, that the assessment of entailment types is time-consuming and unreliable, when based on a direct comparison of complex text clauses and paraphrase sets. We therefore simplify the task, breaking it down into two steps: First, annotators were asked to assign semantic relations between the event descriptions in the paraphrase sets and their instantiations in narrative texts *on the lexical level only*, labeling the event-denoting verbs and the participant-denoting noun phrases, as is described in this section. Second, we automatically derive an approximate clause-level entailment type from the lexical-level labels in a quasi-compositional way from the manually annotated lexical entailment relations, as addressed in Section 4.

In the following subsections, we describe the lexical entailment annotation we conducted on the DeScript and InScript data for events/verbs (Section 3.1.) and participants/nouns (Section 3.2.).

3.1. Events

To prepare the data for the event annotation, we made use of the gold alignment (cf. Figure 3): Each event-denoting verb in InScript was presented with the corresponding paraphrase set in DeScript.

Comparing *every* ED in the paraphrase set to the clause in the text is cumbersome due to the number of EDs (up to 76) in each paraphrase set. We therefore simplified the event paraphrase sets by building equivalence classes of EDs that use the same head verb, which we called *event patterns*. They are derived semi-automatically by summarizing EDs that contain the same main verb, and replacing noun phrases with their participant type label (upper part of Figure 4). Also, participants that do not appear in every ED are put in brackets to mark them as optional.

In the annotation process, annotators were only shown the patterns instead of the full paraphrase set. In the lower part of Figure 4, the verb labeled as DIG is compared to the event patterns extracted from the DIG paraphrase set. Verbs were presented in their sentential context and highlighted.

Instead of annotating each pattern, the guidelines required annotators to select only the most similar pattern for the event-denoting verb, and to do the annotation only for this pattern. While this selection results in a non-exhaustive annotation, no important information is lost: The procedure of selecting the most similar pattern retains the minimal inference steps required for the alignment.

After selecting a pattern, annotators were instructed to assess the relation between the verb in DeScript and the verb in InScript and use the context only for lexical disambiguation, i.e. not to assess clause-level entailment. In our annotation schema, we include the following labels:

- *Identity, Synonymy, Hyponymy, Hypernymy*. Defined as in *WordNet* (Fellbaum, 1998). *Hyponymy* describes the case in which the verb in the text is more specific than in the pattern, *Hypernymy* the opposite.

- *Incorporation*. One verb includes a participant, which is explicitly mentioned with the other verb.
  Example: I ***sprinkled flour*** *in the pan.* – ***flour*** CAKE_TIN

- *Diathesis*. This covers active/passive alternation (Ex. 1) and verbs that are conceptually equivalent but have different syntactic realizations. In *FrameNet* (Ruppenhofer et al., 2006), these verbs would typically be associated with the same frame (Ex. 2).
  Example 1: *The cake* ***was cut****.* – ***cut*** CAKE
  Example 2: *The cake* ***went*** *in the preheated oven* – ***put*** *(*CAKE*) in* OVEN

- *Phrasal Verb.* One of the verbs is a particle verb that has the same meaning as the other verb.
  Example: ***I went out*** *to the grocery store.* – ***go*** *to* STORE

- *Inference.* A complex inference is needed to associate the verbs with the same event.

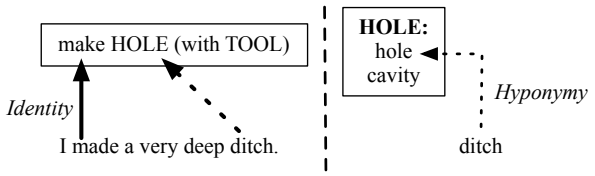

Figure 5: Aligning participants with a pattern (left), selecting a noun from the participant paraphrase set and labeling (right).

Example 1: *I **put** the decorations on the cake. - **use*** DECORATION
Example 2: *I **made sure** that I had all the ingredients that I needed. - **gather*** INGREDIENTS

- *NoMatch.* No match is possible. This is typically the case when annotation errors occur in InScript.

3.2. Participants

Associating verbs in a textual mention with the ED verb is straightforward, since there is only one head verb on each side. Linking participants in contrast requires additional annotation effort: They can be left out or appear more than once in one clause. We thus divided the participant annotation into two steps:

Participant Alignment. In order to distinguish missing from realized participants, annotators had to align all participants that are relevant to the event in question in the text with their counterparts in the pattern, i.e. with the participant type labels. In Figure 5, this step corresponds to the dotted arrow on the left side. Participants that were mentioned in the text and that play a role in the event, but that do not have a counterpart in the ED, were marked as *Additional*. Required participants in the pattern that were not aligned were automatically labeled as *Missing* afterwards.[1] As an example for this label, consider the text clause *I used a shovel*, with the matching pattern *use TOOL for HOLE*. The participant HOLE is omitted, but mentioned as mandatory in the pattern, i.e. it appears in every ED of the respective paraphrase set (*use shovel for making a **hole*** and *use shovel for **hole***). Therefore, it is marked as *Missing* in the pattern.

Lexical Entailment Annotation. To find the appropriate type of lexical entailment for the realized participants, annotators were then shown the participant paraphrase sets from DeScript for all aligned participants. Just as for event patterns, annotators had to choose the best matching, most similar noun from the set and assign a lexical entailment label.

As for verbs, the annotation schema includes the relation types, *Identity, Synonymy, Hyponymy, Hypernymy, Meronymy, Holonymy* and *Co-Hyponomy*, and the additional labels *Inference* and *NoMatch*, as defined in Section 3.1. We add the label *Instance*, which is used when the noun in InScript is a proper noun or entity mention of the type expressed by the DeScript noun (e.g. *number 77 – bus*).

The right side of Figure 5 illustrates this part of the annotation: The noun "ditch" (which was previously aligned with the pattern) is compared to all participant descriptions for HOLE, linked to the lexical description "hole" and labeled with *Hyponymy*. Figure 8 shows the fully labeled instance with participant and event annotations.

To simplify the annotation, we make the assumption that each noun has only one sense per scenario: In the PLANTING A TREE scenario, the polysemous word *stem* e.g. always describes a part of a tree. In order to reduce the annotation effort, we presented all different noun types per participant type only once, rather than every single mentioned token in its sentential context. This *on sense per scenario* assumption is similar to the *one sense per discourse* hypothesis, which is often used in word sense disambiguation models (Gale et al., 1992).

[1]The protagonist of the story, being very rarely mentioned in the EDs, was excluded from the alignment.

Entailment Type	**Labels**
Identity $=$	Identity
Equality $\equiv$	Synonymy, Phrasal Verb, Diathesis
Entailment $\sqsubset$	Hyponymy, Instance, *Additional*
Reverse ent. $\sqsupset$	Hypernymy, *Missing*
Partial Entailment ∞	Inference, Incorporation, Meronymy, Holonymy, Co-Hyponymy
Non-Entailment $\#$	NoMatch

Figure 6: Entailment types.

	$=$	$\equiv$	$\sqsubset$	$\sqsupset$	∞	$\#$
$=$	$=$	$\equiv$	$\sqsubset$	$\sqsupset$	∞	$\#$
$\equiv$	$\equiv$	$\equiv$	$\sqsubset$	$\sqsupset$	∞	$\#$
$\sqsubset$	$\sqsubset$	$\sqsubset$	$\sqsubset$	∞	∞	$\#$
$\sqsupset$	$\sqsupset$	$\sqsupset$	∞	$\sqsupset$	∞	$\#$
∞	∞	∞	∞	∞	∞	$\#$
$\#$	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$

Figure 7: Combination table for lexical entailment classes.

4. Clause-Level Entailment: Composition

In the previous section, we described the lexical entailment annotation on verbs and nouns, i.e. on a sub-event level. In this section, we now explain a method for an automatic, quasi-compositional computation of clausal-level entailment types. We compose the types from the lexical-level entailment labels of the verb and all its annotated noun dependents.

Inspired by the textual inference method used in MacCartney and Manning (2007) and MacCartney and Manning (2009), we compute the type of clause-level entailment between InScript event mentions and DeScript patterns from the manually annotated word-level entailment labels. Following MacCartney, we group these labels according to their truth-conditional effects, and associate each group with one of six entailment types, shown in Figure 6. We adopt four entailment types from the schema of MacCartney and Manning (2009) and add two new types: We extend the schema with *Identity*, which is logically speaking a sub-case of *Equality*. Also, we use *Partial Entailment* to cover all cases of semantic relatedness which do not correspond to a direct entailment type; most prominent are the *Inference* cases.

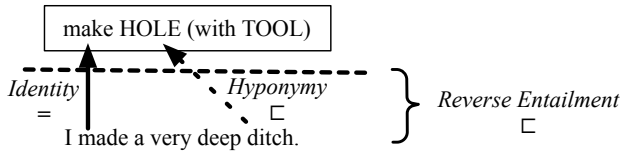

Figure 8: One fully labeled event instance, with compositionally derived clausal entailment label (*Reverse Entailment*).

Cases of *Additional* and *Missing* participants are deletions and insertions in MacCartney's terminology, and therefore have entailment and reverse entailment effects, respectively. To compute the clausal entailment type, we combine the word-level entailment labels for verbs and nouns, according to the composition table in Figure 7. The result of combining two lexical labels is in general the weaker entailment relation of the two[2]. The only exception is the pair $\{\sqsubset, \sqsupset\}$, which results in *Partial Entailment*. The application of the binary computation is commutative, so the clausal entailment type can be read off the set of lexical labels.

Figure 8 shows the running example with all lexical entailment annotations. The set of word-level entailment labels is $\{=, \sqsubset\}$, given the lexical entailment labels {*Identity, Hyponymy*}. The clausal entailment type is the weakest lexical type, i.e., $\sqsubset$ (*Reverse Entailment*).

5. Annotation Statistics

The annotation was performed on all 280 texts of the InScript corpus addressing one of the three selected scenarios. Annotation was done by two native speakers of German with a good command of English. A total of 3,427 verb mentions and 1248 noun types was annotated, respectively. We used SWAN[3] (Gühring et al., 2016) for the annotation.

5.1. Lexical Level

5.1.1. Inter-Annotator Agreement

For both verb and participant annotation, agreement is computed on two levels: First, we report how often the same pattern or head noun was selected. Second, in cases where the same pattern/noun was selected, we report the label agreement. For the verb annotation, annotators chose the same pattern in 82.4% of cases. For participants, the annotators chose the same realization from DeScript in 74.3% of cases.

On verbs, the annotators agreed on the label with $\kappa = 0.72$ (*substantial agreement*, (Landis and Koch, 1977)). On participants, they agreed with $\kappa = 0.742$ (*substantial agreement*).

Not every case of disagreement is critical: In many cases there is more than one plausible solution. In the PLANTING A TREE scenario, for example, the word *shovel* could be interpreted either as a *Synonym* or *Co-Hyponym* of *spade*. There are also no sharp boundaries between classes. In particular, annotators had difficulties with annotating *Inference* cases. Therefore, we decided not to adjudicate the annotation, but average over the distributions for evaluation.

[2]The list $=, \equiv, \{\sqsubset, \sqsupset\}, \infty, \#$ orders the labels from strongest to weakest.

[3]https://github.com/annefried/swan

Label	Events	Participants
Identity	58%	76%
Synonymy	5%	6%
Hyponymy	5%	5%
Phrasal Verb	5%	-
Inference	13%	1%
NoMatch	7%	7%
Other	7%	5%

Table 1: Distribution of lexical labels on events and participants.

5.1.2. Label Distribution

Table 1 gives label distributions for participants and events on the lexical level, averaged over both annotations. As mentioned before, we annotated each noun type only once per participant type rather than annotating every mention of a noun separately. To compute the numbers depicted in Table 1, we copy the type-level annotation to every single appearance of the noun, to give a better idea of the actual distribution.

For both verbs and participants, the most frequent relation chosen by both annotators is *Identity*. Among the lexical relations, *Hyponymy* is more frequent than *Hypernymy*, which is consistent with the expectation that concrete event/participant mentions use more specific verbs/nouns than the abstract descriptions in the script knowledge base (cf. Modi et al. (2016)). *Diathesis*, *Incorporation* and *Hypernymy* for verbs, and *Meronymy*, *Hypernymy*, *Holonymy*, *Instance* and *Co-Hyponymy* for participants appear only very rarely and are subsumed under *Other* in the table.

5.2. Clausal Level

Figure 9 shows the distribution of the resulting clausal entailment labels for both annotators. *Identity* makes up for the largest part of cases (38 %), illustrating the high lexical coverage of crowdsourced script representations.

Entailment cases are significantly more frequent than *Reverse Entailment*, which is in line with the leading assumption that an event mention should entail a description of its event type, and with the observation that event-denoting clauses usually use longer sentences and more specific vocabulary than event descriptions given by the script knowledge base.

Both *Entailment* and *Reverse Entailment* mainly contain cases in which the participant is not realized on one side, and are only rarely composed from *Hypernymy* or *Hyponymy* cases. A typical example from the TAKING A BUS scenario is the text clause *wait for several minutes*. The most similar ED in the scenario is just *wait*, so the time expression is an additional participant and thus results in a clause-level entailment.

There is a large number of *Partial Entailment* cases (20%), which are in many cases composed of one or several *Inference* labels. One typical example of such a case from the TAKING A BUS scenario is the text clause *the driver pulled over*. The paraphrase sets only contain phrases like *bus stops* or *arrive at destination*. In this case, a system would need to know that *pull over* in the bus context is a paraphrase for

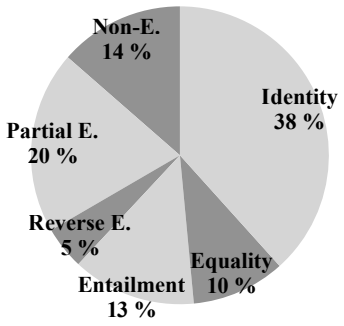

Figure 9: Distribution of composed labels.

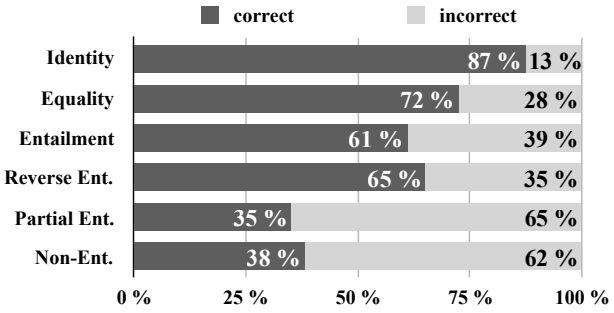

Figure 10: Performance of Ostermann et al. (2017) on clausal entailment classes.

stop, which requires contextual inference. These cases also appear in other scenarios, e.g. in the PLANTING A TREE scenario: *look at trees* is a contextual paraphrase of *choose a tree*. This can be seen as an indicator for the difficulty of the text-to-script mapping task, but it also indicates that the script resource is not exhaustive enough to contain all possible formulation variants for events.
Lastly, we found that a large number of *Non-Entailment* cases are derived from annotation errors in InScript.

6. Entailment Type and System Performance

The previous section investigated the difficulty of the task of text-to-script mapping itself. In this section, we now look at the performance of the only published model for text-to-script mapping and show that the cases we identified as complex are indeed most challenging for the model. We apply the entailment-type annotation to Ostermann et al. (2017)'s text-to-script mapping system, breaking down its event-labeling performance on our annotated InScript subcorpus to clause-level entailment classes. The results for all script-relevant clauses are shown in Figure 10.
Identity (verb and all head nouns are lemma-identical) is easiest to model, and so as expected provides the best results. That the accuracy is not 100% is due to the fact that sometimes the same verb lemma occurs in different paraphrase sets. This holds in particular for light verbs such as *get*, which can be used to instantiate many different events throughout a scenario. In the BAKING A CAKE SCENARIO, for example, *get* appears in the CHOOSE_RECIPE paraphrase set (*get your recipe*), as well as in GET_INGREDIENTS (*get a box of cake mix*), GET_UTENSILS (*get a pan*), etc.
In general, the important result is not the accuracy of the identity case in itself, but in connection with the high percentage of identity cases (38%). The high number of realization variants for event mentions contained in the paraphrase sets carry out a large part of the mapping task.
To illustrate the positive effect of a large paraphrase set on system performance, we compare the result with a situation in which there is only one ED per paraphrase set. To this end, we picked one ED randomly from the paraphrase sets (average size is 25, using a token-based count), as representative(s) of the event type, and then computed the number of verb identity cases automatically: They would drop from 58% to 26.9%. With a similar drop for participant identity cases, the resulting clause-based percentage of identity cases would be below 20% (instead of 38%), probably leading to a dramatic drop in labeling accuracy.
Partial entailment is the class that is most difficult to model. This result provides a complementary message about the script knowledge base. About 80% of the clause-level *Partial Entailment* cases contain a word-level *Inference* relation, which is a strong indicator of the need for methods able to handle more complex inference: Crowdsourced collections of linguistic realization variants help to avoid complex inference in many cases, but cannot completely replace it. This emphasizes the need for larger script data collections that cover even more description variants for events and participants.
The low performance on the *Equality* cases is mainly due to the fact that it subsumes the difficult *Diathesis* and *Phrasal Verb* relations. In contrast, the results for *Reverse Entailment* are better than one would expect: This is mostly due to the fact that most *Reverse Entailment* cases (approx. 85%) consist of combinations of *Equality* or *Identity* with an *Unrealized* participant. Even if one participant is unrealized, the correct label is mostly identified. For the text clause *I placed the tree*, *hole* is a required participant, occurring in every ED, but nevertheless the overlap is high enough to select the correct label. The more difficult *Hypernymy* cases make up only 15% of the *Reverse Entailment* cases. The situation is similar for the *Entailment* cases. Here, the simpler *Additional* participant cases make up 50%.
Finally, the model has low accuracy for the *Non-Entailment* cases. However, it is substantially above the random baseline. A possible reason is that the compositional computation of the clause-level entailment type amounts to a generalization to the worst case. Thus a number of pairs end up in the *Non-Entailment* class although there is only a minor local incompatibility.

7. Background and Related Work

Our work is based on script representations in which events are encoded as paraphrase sets. There also exist other research directions on modeling script knowledge. The most prominent alternative representation of script knowledge is that of *narrative chains*, proposed by Chambers and Jurafsky (2008) and subsequently extended by Chambers and Jurafsky (2009), Pichotta and Mooney (2014) and (Ahrendt and Demberg, 2016), to name but a few. Narrative chains

have been used for event prediction (Chambers and Jurafsky, 2009; Pichotta and Mooney, 2014; Rudinger et al., 2015; Modi et al., 2016) or the related *story cloze* task (Mostafazadeh et al., 2016; Pichotta and Mooney, 2016), in which complete sentences are predicted.

Narrative chains (and their aforementioned extensions) differ in two relevant aspects from the script representations used in our study. Instead of using paraphrase sets, events are represented as typed dependency relations between a verb and one of its dependents (the *protagonist*). Another difference is that narrative chains are intended to be learned automatically from large collections of unannotated text. By contrast, the script representations used in our study are learned from crowdsourced sequences of event descriptions, which are more focused and more detailed compared to narrative chains: They also contain events which are often not mentioned in text, since they are assumed to be background knowledge (Chambers, 2017). These two differences imply that the results of our annotation cannot easily be transferred to script representations along the lines of Chambers and Jurafsky (2008).

Text-to-script mapping is similar to the task of *recognizing textual entailment* (*RTE*, Dagan et al. (2006)), in which systems have to decide whether a *text* entails a *hypothesis*. The text entails the hypothesis if a human reader would infer from the text that the hypothesis is most likely true. In our case, event mentions and event descriptions correspond to texts and hypotheses, respectively. The lexical entailment annotation in our study is inspired by similar annotation efforts in the context of RTE, for instance Garoufi (2007), who used lexical entailment annotations to annotate the RTE-2 data set (Bar Haim et al., 2006). The label set we use is inspired by their set of lexical entailment categories, and they also conducted an alignment step similar to our *participant alignment*. Our computational derivation of clausal-level labels is built on MacCartney and Manning (2007), MacCartney and Manning (2009), and MacCartney (2009).

8. Conclusion

In this work, we annotated event mentions in narrative texts with semantic relations that need to be modeled when mapping the mentions to script events that are represented as paraphrase sets. We provide a lexical-level entailment annotation between event-denoting verbs and participant-denoting nouns of narrative texts on the one side, and event and participant descriptions of a script on the other side. We then derive clause-level entailment labels that highlight the coverage of crowdsourced paraphrase sets associated with event types, as compared to the textual variation in naturalistic texts. We find that script representations in the form of paraphrase sets can cover a large number of description variants of an event in a text. However, the alignment of a substantial amount of event mentions requires a deeper inference of multiple semantic relations.

Based on our annotation, we analyze the performance of an existing text-to-script mapping system on the different entailment classes. The results indicate (1) that paraphrase sets as constitutive part of script representations can massively increase the accuracy of text-to-script mapping systems and (2) that the tested model is mostly unable to account for the more complex cases.

The data set is available at `http://www.sfb1102.uni-saarland.de/?page_id=2582`.

Acknowledgements

We would like to thank the anonymous reviewers and Michael Roth for their helpful comments on the paper. Also, we thank our student assistants Sophie Henning, Sarah Mameche and Leonie Harter for their help with the annotations. This research was funded by the German Research Foundation (DFG) as part of SFB 1102 'Information Density and Linguistic Encoding'.

9. Bibliographical References

Ahrendt, S. and Demberg, V. (2016). Improving event prediction by representing script participants. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 546–551.

Bar Haim, R., Dagan, I., Dolan, B., Ferro, L., Giampiccolo, D., Magnini, B., and Szpektor, I. (2006). The second PASCAL Recognising Textual Entailment Challenge.

Chambers, N. and Jurafsky, D. (2008). Unsupervised learning of narrative event chains. *Proceedings of ACL-08*.

Chambers, N. and Jurafsky, D. (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*.

Chambers, N. (2017). Behind the scenes of an evolving event cloze test. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*, pages 41–45.

Dagan, I., Glickman, O., and Magnini, B. (2006). The PASCAL Recognising Textual Entailment Challenge. *Quiñonero-Candela, J.; Dagan, I.; Magnini, B.; d'Alché-Buc, F. (Eds.) Machine Learning Challenges. Lecture Notes in Computer Science*, 3944:177–190.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. MIT Press.

Gale, W. A., Church, K. W., and Yarowsky, D. (1992). One sense per discourse. In *Proceedings of the Workshop on Speech and Natural Language*, pages 233–237. Association for Computational Linguistics.

Garoufi, K. (2007). Towards a better understanding of applied textual entailment. Master's thesis, Saarland University, Saarbrücken, Germany.

Gühring, T., Linz, N., Theis, R., and Friedrich, A. (2016). SWAN: an easy-to-use web-based annotation syste. In *Proceedings of the 13th Conference on Natural Language Processing (KONVENS). September 19-22, Bochum, Germany*.

Landis, J. R. and Koch, G. G. (1977). The Measurement of Observer Agreement for Categorical Data. *Biometrics*, 33(1):pp. 159–174.

MacCartney, B. and Manning, C. D. (2007). Natural logic for textual inference. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 193–200. Association for Computational Linguistics.

MacCartney, B. and Manning, C. D. (2009). An extended model of natural logic. In *Proceedings of the eighth international conference on computational semantics*, pages 140–156. Association for Computational Linguistics.

MacCartney, B. (2009). *Natural Language Inference*. Ph.D. thesis, Stanford University.

Modi, A. and Titov, I. (2014). Inducing Neural Models of Script Knowledge. In *Proceedings of the Conference on Computational Natural Language Learning (CoNLL)*, Baltimore, MD, USA.

Modi, A., Anikina, T., Ostermann, S., and Pinkal, M. (2016). InScript: Narrative texts annotated with script information. *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 16)*.

Modi, A., Titov, I., Demberg, V., Sayeed, A., and Pinkal, M. (2017). Modelling Semantic Expectation: Using Script Knowledge for Referent Prediction. *Transactions of the Association for Computational Linguistics*, 5:31–44.

Mostafazadeh, N., Chambers, N., He, X., Parikh, D., Batra, D., Vanderwende, L., Kohli, P., and Allen, J. (2016). A Corpus and Evaluation Framework for Deeper Understanding of Commonsense Stories. *Proceedings of NAACL-HLT 2016*, pages 839–849.

Ostermann, S., Roth, M., Thater, S., and Pinkal, M. (2017). Aligning Script Events with Narrative Texts. *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*, pages 128–134.

Pichotta, K. and Mooney, R. J. (2014). Statistical Script Learning with Multi-Argument Events. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, volume 14, pages 220–229.

Pichotta, K. and Mooney, R. J. (2016). Using Sentence-Level LSTM Language Models for Script Inference. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 279–289.

Regneri, M., Koller, A., and Pinkal, M. (2010). Learning Script Knowledge with Web Experiments. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 979–988, Stroudsburg, PA, USA. Association for Computational Linguistics.

Rudinger, R., Rastogi, P., Ferraro, F., and Durme, B. V. (2015). Script Induction as Language Modeling. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1681–1686.

Ruppenhofer, J., Ellsworth, M., Petruck, M. R., Johnson, C. R., and Scheffczyk, J. (2006). FrameNet II: Extended theory and practice.

Schank, R. C. and Abelson, R. P. (1975). *Scripts, plans, and knowledge*. Yale University New Haven, CT.

Wanzare, L. D. A., Zarcone, A., Thater, S., and Pinkal, M. (2016). DeScript: 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)*.

Wanzare, L. D. A., Zarcone, A., Thater, S., and Pinkal, M. (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*.