Supervised Relation Classification as Two-way Span-Prediction

Amir DN Cohen AMIRDNC@GMAIL
Bar Ilan University

Shachar Rosenman SHACHAROSN@GMAIL
Bar Ilan University

Yoav Goldberg YOAV.GOLDBERG@GMAIL
Bar Ilan University & Allen Institute for AI

Abstract

Most of the current supervised relation classification algorithms use a single embedding to represent the relation between a pair of entities. We argue that a better approach is to treat the relation classification task as a Span-Prediction problem, similar to Question Answering. We present a span prediction based system for relation classification and evaluate its performance compared to the embedding-based system. We demonstrate that the supervised span prediction objective works significantly better than the standard classification-based objective. We achieve state-of-the-art results on the TACRED, SemEval task 8, and CRE datasets.

1. Introduction

The relation extraction (RE) task revolves around binary relations (such as "[e_1] founded [e_2]") that hold between two entities. The task is: given a corpus and a list of semantic relations, return entity pairs e_1, e_2 that are connected by one of the predefined relations. This is often posed as a Relation Classification task (RC), in which we are given a sentence and two entities (where each entity is a span over the sentence), and need to classify the relation into one of $|R|$ possible relations or to a null "no-relation" class if none of the $|R|$ relations hold between the given entities. RE datasets, including the popular and large TACRED dataset [Zhang et al., 2017], all take the relation classification view, by providing tuples of the form (s, e_1, e_2, r), where s is a sentence, e_1, e_2 are entities in s and r is a semantic relation between e_1 and e_2. Consequently, the state-of-the-art models follow the classification view: the sentence and entities are encoded into a vector representation, which is then being classified into one of the $|R|$ relations. The training objective then aims to embed the sentence + entities into a space in which the different relations are well separated. We argue that this is a sub-optimal training architecture and training objective for the task, and propose to use span-predictions (SP) models—as used in extractive question-answering—instead.

Our method can be summarised as follows: convert each sample in the RC datasets into several new SP subsamples, where each of the subsamples is added with a predefined semantic indicator that represents a specific relation (e.g. *"per::date_of_birth X"* or *"When was X born?"*). Then, train a dedicated SP model on these subsamples. For inference, split each test sample into subsamples as before, evaluate each of them independently and

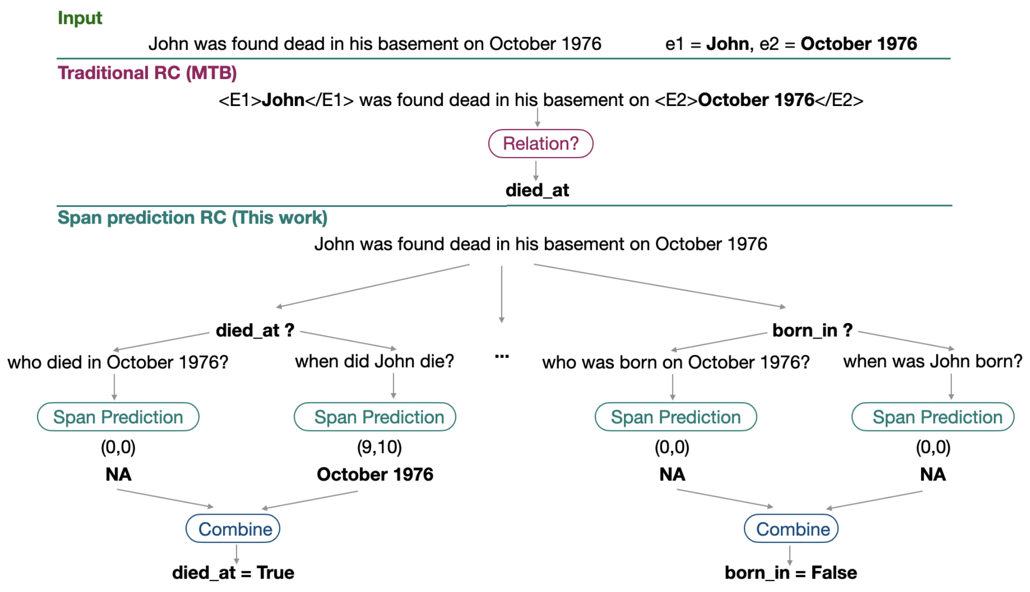
Input
John was found dead in his basement on October 1976
e1 = John, e2 = October 1976
Traditional RC (MTB)
<E1>John</E1> was found dead in his basement on <E2>October 1976</E2>
Relation?
died_at
Span prediction RC (This work)
John was found dead in his basement on October 1976
died_at ?
who died in October 1976?
when did John die?
...
born_in ?
who was born on October 1976?
when was John born?
Span Prediction
(0,0)
NA
(9,10)
October 1976
Combine
died_at = True
born_in = False

Figure 1: Traditional RC (top) VS our span-prediction approach (bottom). for each relation type that is compatible with the marked entity type, we create two questions. If the model answers one of them correctly, we assert the relation over the two entities.

aggregate the result to return a prediction for the entities relation type. We show a high-level flow of this procedure in Figure 1.

Interestingly, even when the model is exposed to tens of thousands of samples during training, the added semantic information in the templates gives a significant boost to the model, increasing its accuracy by $1.8F1$.

Alt et al. [2020] analyzed the errors of current RC systems and estimated that most errors originate from incorrectly predicting "no relation" (approx. 63.5%) and by considering wrong arguments in the input sentence (approx 10.7%). Our model is specially tailored to minimize these errors. We demonstrate this on the TACRED and SemEval datasets. Our method surpasses the current state-of-the-art on these datasets by $2.3F_1$ points on TACRED and $0.9F_1$ points on SemEval. Additionally, we experiment with the newly released "challenge relation extraction" (CRE) dataset, which was made specifically to test the existence of shallow heuristics in RE models. we surpass current state-of-the-art models by $5.0F1$. On all three datasets, our span-prediction models outperform existing RC methods.

To summarize our contribution, while all current approaches to supervised relation classification use an embedding-based technique, we propose a span-prediction-based one, which significantly improves state-of-the-art scores on several well-known datasets.

2. Related Work

The framework of question answering was used to solve a variety of NLP tasks like coreference resolution [Wu et al., 2019], event extractions [Du and Cardie, 2020], nested named entities [Li et al., 2019a], multi-turn entity extraction [Li et al., 2019b], and others [Jiang et al., 2019].

Levy et al. [2017] suggested using QA for *zero-shot RE/RC* by framing each relation by a predefined question. Our work can be seen as the *fully supervised* variation of their work: we show that even in the supervised case, moving to a span-prediction formulation is worthwhile, and that the gains stem from the formulation more than from the knowledge obtained from the QA dataset. Indeed, when fine-tuning our span-prediction RC from a SQuAD-based model, we get *worse* results than when random initializing it.

Another key work in this area is the work of Wei et al. [2019], who proposed to use span prediction method to improve RE for a corpus with more than one relation in the same sentence. They do this by identifying head entity spans, and then using SP for predicting the tail for each span, along with its relation. Our work focuses on RC instead of RE, and is restricted to answering yes/no judgments about given *(relation1,entity1,entity2)* tuples. While the methods cannot be compared directly, we did attempt to evaluate Wei et al's method on an RC dataset, by employing it as an RE model and looking for how many of the relations in the RC datasets were recovered. This yielded very low recall scores of 0.24 on TACRED and 0.71 on semEval[1].

While the mentioned works used SP models to improve performance on a specific task, it is worth mentioning that other works have used QA for different reasons, like [He et al., 2015] that used QA as an easier way to annotate data for the SRL task.

3. Embedding Classification vs Span-Prediction

Embedding-Based Relation Classification A *RC sample* takes the form (c, e_1, e_2, r) where $c = [c_0, \ldots, c_n]$ is a context (usually a sentence), e_1 and e_2 are spans that correspond to head and tail entities and are given as spans over the sentence, and $r \in R \cup \{\emptyset\}$ is a relation from a predefined set of relations R, or $\emptyset$ indicating that no relation from R holds. *RC classifier* takes the form of a multi-class classifier:

$$f_{rc}(c, e_1, e_2) \mapsto r \in R \cup \{\emptyset\}$$

The training objective is to score the correct $r \in R \cup \{\emptyset\}$ over all other incorrect answers, usually using a cross-entropy loss. State-of-the-art methods [Baldini Soares et al., 2019] achieve this by learning an embedding function $embed(c, e_1, e_2)$ that maps instances with the same relation to be close to the embedding of the corresponding relation in an embedding space. The embedding function is based on pre-trained masked LMs such as SpanBERT [Joshi et al., 2020], RoBERTa [Liu et al., 2019] and ALBERT [Lan et al., 2019].
Span Prediction A *SP sample* takes the form of (c, q, e_a) where $c = [c_0, \ldots, c_m]$ is a context (a sentence or a paragraph), $q = [q_0, \ldots, q_l]$ is a query, and e_a is the answer to the query

1. We tried to make the tasks more similar by skipping the head-span prediction step and feeding the algorithm with the gold head entity in the sentence, and letting it infer the relevant relations and their tails. This yielded also similarly low recall.

represented as a span over c, or a special out-of-sequence-span indicating that the answer does not exist.[2]
SP model takes the form of a *span predictor* from a c, q pair to a span over c:

$$f_{qa}(c, q) \mapsto e_a \in [0 \dots m] \times [0 \dots m]$$

This predictor takes the form:

$$\arg\max_{e_a} s_{c,q}(e_a)$$

where $s_{c,q}(e_a)$ is a learned span scoring function, and e_a ranges over all possible spans. The training objective is to maximize the score of the correct spans above all other candidate spans. The scoring function in state-of-the-art models [McCann et al., 2018, He et al., 2015, Wu et al., 2019] also make use of pre-trained LMs.

3.1 Method Comparison

The question q ("where was Sam born?") in QA can be thought of as involving a span e_q ("Sam") and a predicate r_q ("where was born"). Under this view, the SP classifier can be written as:

$$f_{qa}(c, e_q, r_q) \mapsto e_a$$

compared to the relation classifier:

$$f_{rc}(c, e_1, e_2) \mapsto r$$

Note that both methods include a context, two spans, and a relation/predicate, but the RC models classify from two spans to a relation (from a fixed set), while the SP model classify from a span and a relation (from a potentially open set) into another span.

While both the traditional and SP methods embed the input prior to classification, the items that are being embedded in each, change. In traditional RC the embedding h_{re} is based on the context and entities:

$$h_{re} = embed(c, e_1, e_2)$$

while for SP RC the embedding h_{qa} encodes both the context and the question (the relation of interest and one of the entities):[3]

$$h_{qa} = embed(q, c) = embed(r, e_1, c)$$

Note that the SP embedding includes the relation name, as well as template words that surround the (r, e_1) pair. This has several benefits, as we explain below.[4]

2. In practice, this span is the out-of-sentence *CLS* token.
3. In practice, the embedding is obtained via a pre-trained LM such as BERT, and as per-usual is prefixed with a `CLS` token, while the different components are separated with a `SEP` token.
4. Another way to encode input for the SP model is by encoding the full sentence and adding the focus for each relation in the final layer of classification (as used by Wei et al. [2019]): $h_c = embed(c)$ While this representation considers less information than the previous one (resulting in lower accuracy), it can be used to classify multiple entities and relation with one calculation, which make it more computationally efficient.

3.2 Implications

Relation type indication for the pretrained model. The inclusion of the relation r in the input to the contextualized embedder allows the embedder to specialize on a specific relation. For example, consider the sentence "*Martha gave birth to John last February*". The entity John participates in two relations: "date of birth" and "parents of". The RC embedding will have to either infer the relation based on the entities, or else preserve information regarding both relations, while in the span-prediction case the embedding takes the relation r into account, and can focus on the existence (or nonexistence) of one of the entities as the argument for this relation. Focusing on a specific relation in the embedding stage (which involves most of the computation of the model) allows using all of the model computation for a specific relation.

Sharing of semantic information. The span-prediction model is based on templates encoding r and e, and these templates may pass valuable information to the model: (1) by containing semantic information that is correlated to the target relation (e.g. questions that represent the relation); and (2) by containing information that can help generalize over different relations.

For example, consider the relation "born in" with the template question "Who was born in X?" and the relation "parent of" with the question "Who is the parent of X?". While the relations are different from each other, they both contain an entity of type "person", a similarity which is communicated to the model by the use of the shared word "Who". This can help the model generalize commonalities across relation types when needed. While the template input might look insignificant in the supervised setting, where training data is abundant, in practice it has a significant effect on the overall model performance, as shown in Section 5.2.

More demanding loss function. During training, relation-classification models classify sentences with marked entities to one of $|R| + 1$ relation types. Span prediction models are also required to decide whether the sentence contains a given relation (they should predict if the sentence contains the answer or not), but they are also required to predict the span of the missing argument. This means that the span-prediction models are required to predict the relation between the input entities in addition to the relation itself.

Limitations. It is important to note, however, that the span-prediction method is more computationally expensive: instead of performing a single contextualized embedding operation followed by $k + 1$-way classification, we need to perform k contextualized embedding operations (and in our case, $2k$ such operations), each of them followed by the scoring of all spans. We leave ways of improving the computational efficiency of the model to future work.

4. Method: Reducing RC to span-prediction

Given the uncovered similarity between RC and span-predicting showed in Section 3.1, we now describe how to reduce RC to SP.

Given an RC instance $(c, e_1, e_2) \mapsto rel$ we can create an SP instance $(c, q = (e_q, rel_q)) \mapsto e_a$ as follows. Let $T_{rel}(e)$ be a *template function* associated with relation rel. The function takes an entity e and returns a question. For example, a template for date-of-birth relation might be T_{dob} ="*When was __ born*", and $T_{dob}(\text{Sam})$ ="*When was Sam born?*". Given an RC instance $(c, e_1, e_2) \mapsto rel$ we can now create a span-prediction instance $(T_{rel}(e_1), c) \mapsto e_2$,

RC sample	**Relation candidates**	**Question (Reverse Question)**	**Answer**
John was born on **1991**	"Date of birth"	When was John Born?	1991
		(Who was born on 1991?)	John
	"Date of death"	When did john die?	N/A
		(Who died on 1991?)	N/A
Mary is **John**'s employer	"employer of"	Who is employed by Mary?	John
		(Who is John's employer?)	Mary
	"siblings"	Who is the sibling of Mary?	N/A
		(Who is the sibling of John?)	N/A
	"parents of"	Who is the child of Mary?	N/A
		(Who is the parent of John?)	N/A

Figure 2: **Supervised dataset construction**. Example of span-prediction samples that are generated from RC samples. The RC sample contains the sentence, entities (in bold), and relation, while the span-prediction sample has a context (same as the sentence in the RC sample), a query, and an answer. A set of relation questions are created based on the RC entities types.

and return that the relation *rel* holds if the span returned from $f_{qa}(T_{rel}(e_1), c)$ is compatible with e_2. This is essentially the construction of Levy et al. [2017]. We extend it as follows:

Bidirectional questions. We note that the decision to predict e_2 based on e_1 is arbitrary, and that the opposite direction can also be used using the template "Who was born on __?", to predict e_1 from e_2.

We propose to use both options, by associating a relation *rel* with *two* templates, $T_{rel}^{e_1 \to e_2}$ and $T_{rel}^{e_2 \to e_1}$, creating the two corresponding SP instances, and combining the two answers. Concretely, given the RC instance:

RC:$(c, Sam, 1991) \mapsto$ *date-of-birth*

we create the two SP instances:

QA1:$(c,$ *When was Sam born?*$) \mapsto$ *1991* QA2:$(c,$ *Who was born in 1991?*$) \mapsto$ *Sam*

We show in Section 5 that using two questions indeed results in substantial improvements.

Template formulation. Note that while in this example we formulate the questions in English, a simpler template might also be used. We also experiment with a template that replaces the question by the relation name and another template that used an unused token for each relation. We elaborate on the template variations in detail in Section 5.

Answer combination. There are various possible strategies to combining the two answers. An approach which we found to be effective is to combine using an OR operation: if either of the returned spans is compatible with the expected span,[5] the relation *rel* is returned, and if neither of them is compatible, the answer is no-relation.

A natural alternative is to combine using an AND operation, requiring the answers of the two questions to be compatible in order to return *rel*. In our experiments (Section 6), this yielded lower F-scores on the relation classification task, as we classified more cases as no-relation when we shouldn't have. The span predictor network had an easier time answering one formulation on some instances, and the other formulation on others. As

5. Two non-empty spans are said to be compatible if either of them contains the other.

span-prediction model quality improves, future applications may reconsider the combination method.

Binary vs. Multiclass. The above reduction targets a binary version of RC, where the relation is given and the classifier needs to decide if it holds or not. We extend it to the multi-class version by creating a version for each of the relevant[6] relations.[7]

Supervised dataset construction. The reduction allows us to train a SP model to classify RC instances. For each RC training instance (c, e_1, e_2, r), where $r \in R \cup \{\emptyset\}$, we consider all relations $r' \in R$ which are compatible with (e_1, e_2).[8] We then generate two SP instances for each of the compatible relations. Instances that are generated with the templates of the gold-relation r are marked as positive instances (their answer is either e_1 or e_2, as appropriate), while instances that are generated from $r' \neq r$ are negative examples (their answer is the no-answer span). Figure 2 provides an example.

Per-template thresholds. General purpose SP models use a global threshold τ to distinguish between answerable and non-answerable questions given a context. The model output given input sample s is:

$$pred(s) = \begin{cases} \text{e} & \text{if } score(e) - score(no\text{-}answer) > \tau \\ \text{“NA”} & \text{else} \end{cases}.$$

Where e is the token with the highest score and *no-answer* is the no answer span. In the supervised relation classification case, the set of questions is fixed in advance to $2|R|$. We observe that the optimal threshold value for each question is different. We thus set a different threshold value τ^i_{rel} for each template. The threshold is set by converting each labelled sample s into the tuple (v_s, a_s), where $s_v = score(e_s) - score(no\text{-}answer)$ and a_s is a label that equal to 1 iff the sample has an answer or not. To find a relation specific threshold τ_r use the following equation:

$$best\_t(D_r) = \underset{\tau}{argmax} \sum_{s \in D} threshold(s, \tau).$$

Where D_r is a dataset subset containing subsamples with a template based on relation r:

$$threshold(s, \tau) = \begin{cases} a_s & v_s \geq \tau \\ 1 - a_s & \text{else} \end{cases}.$$

5. Main Results

5.1 Datasets and Models

Datasets. We compare ourselves on three RC datasets:

TACRED [Zhang et al., 2017] is currently the most popular and largest RC dataset. It spans 41 “classic” RC relations, which hold between persons, locations, organizations, dates, and so on (e.g, “siblings”, “dates of birth”, “subsidiaries”, etc). TACRED contains 106,264

6. A relation is relevant for a given pair of entities if the entity types match that of the relation.
7. In the rare case (less than 4%) that our model predicts more than one relation, we return one of them arbitrarily.
8. A relation is compatible with a pair of entities if it is between entities with the same named-entity types.

labeled sentences (train + dev + test), where 20% of the data is composed from the 41 relations and the rest 80% are "no relation" instances.

SemEval 2010 Task 8 (SemEval, Hendrickx et al. [2010]), is a smaller dataset, containing 10,717 annotated examples covering 9 relations, without no-relation examples. SemEval relations are substantially different from those in TACRED, covering more abstract relations such as part-whole, cause-effect, content-container, and so on.

Challenge relation extraction (CRE) Rosenman et al. [2020] showed that current RC models have a strong bias towards shallow heuristics that do not capture the deep semantic relation between entities. For example, classifying an entity pair by the entities type + an unrelated event in the sentence. To show this bias empirically, they created a Wikipedia-based challenge dataset intended to be used only for testing, which contains 3000 manually tagged sentences from the TACRED relation set. Each sentence in the dataset contains two entity pairs that are compatible with the same relation. The evaluation of the CRE is binary — the model goal is to indicate if a given relation is found or not found in the sentence. The dataset was evaluated with both SP and RC models, showing that the former generally outperform the later.

Models. We compare our results to several leading models, reporting the results from the corresponding papers. **MTB** [Baldini Soares et al., 2019] is a state of the art RC model which is based on BERT-large, and which does not involve any additional training material except for the pre-trained LM. MTB's way of creating sentence embedding is the current state-of-the-art, and thus our most direct comparison.[9] **KEPLER** [Wang et al., 2019] This model holds the current highest reported RC results over TACRED. It is a RoBERTa based RC model which incorporates additional knowledge in the form of a knowledge graph derived from Wikipedia and Wikidata and uses MTB for sentence embedding. **LiTian** [Li and Tian, 2020] is the current top-scoring model on the SemEval dataset. It uses a dedicated RC architecture and uses the BERT pre-trained LM.

We train span-predicting models using the architecture described in [Devlin et al., 2018], starting from either the BERT-Large [Devlin et al., 2018] or ALBERT [Lan et al., 2019] pre-trained LMs.[10] BERT-large is used to compare the state-of-the-art model reported in [Baldini Soares et al., 2019] on equal grounds, while ALBERT is a stronger pre-trained LM which is used to show the full capabilities of our approach.[11]

5.2 Template variations.

We convert the TACRED and SemEval training sets to span-prediction form in three ways, representing various amounts of semantic information. From most informative to least, the variations are:

Natural language questions (question) For each RC sample we create two samples, as described in Section 4. The complete template list is available in the supp materials.

9. The same paper reports additional results based on external training data, which is not comparable. However, these results have since been superseded by the KEPLER model.
10. We used the implementations provided by Huggingface [Wolf et al., 2019]. Following previous work, used the Adam optimizer, an initial learning rate of $3e^{-5}$, and up to 20,000 steps with early stopping on a dev-set.
11. We also ran preliminary tests using [Liu et al., 2019] and [Joshi et al., 2020] that showed inferior results compared to ALBERT.

	CRE			TACRED			SemEval		
Model	Acc$_+$	Acc$_-$	Acc	P	R	F$_1$	P	R	F$_1$
RC$_{MTB,BERT}$	70.0	64.8	67.1	-	-	70.1	-	-	89.2
LiTian (sota SemEval)	-	-	-	-	-	-	**94.2**	88.0	91.0
SP$_{token,BERT}$	55.0	75.5	66.4	63.3	**78.4**	70.0	92.8	88.8	90.7
SP$_{relation,BERT}$	66.6	72.1	69.6	67.0	76.0	71.2	91.9	83.1	87.1
SP$_{question,BERT}$	**72.5**	**75.0**	**73.9**	**71.1**	**72.6**	**71.8**	90.7	**93.2**	**91.9**
KEPLER$_{RoBERTa+KG}$	-	-	-	72.8	72.2	72.5	-	-	-
SQuAD$_{ALBERT}$	71.5	78.8	75.3	49.7	**78.9**	57.1	-	-	-
SP$_{token,ALBERT}$	80.9	73.2	76.6	72.2	74.6	73.4	-	-	-
SP$_{relation,ALBERT}$	78.2	79.8	79.1	**74.6**	75.2	**74.8**	-	-	-
SP$_{question,ALBERT}$	**81.2**	**79.5**	**80.3**	73.3	71.8	72.6	-	-	-

Table 1: Evaluation on the three datasets. **CRE.** Span prediction model results on CRE, compared to traditional RC and QA model. RC models are relation classification models and SQuAD models are QA models that were trained on the SQuAD 2.0 dataset. **TACRED.** Supervised results on the TACRED datasets. **Top**: Using BERT. This is a direct comparison to the MTB span-prediction model. MTB F$_1$ is taken from the original paper. SP models (except token) suppress MTB. **Bottom**: Using ALBERT. Here the reference point is KEPLER, the current best performing model on this dataset. All the supervised SP-ALBERT models outperform KEPPLER. **SemEval.** Supervised results on the SemEval datasets.

Relation name (relation) Same as the question dataset, but we replace each of the questions with the relation name, entity, and a marker that indicate if it's a head or tail entity. E.g., the relation RC:$(c, John, CEO) \mapsto$ *per:title* will be represented as the questions: QA1:$(c,$ *per:title t John*$) \mapsto$ *CEO* QA2:$(c,$ *per:title h CEO*$) \mapsto$ *John*

Unique tokens (token) Same as the relation dataset, but we replace the relation name with a new reserved token. E.g., the above *per:title* relation will be represented as the questions: QA1:$(c,$ *r2 t John*$) \mapsto$ *CEO* QA2:$(c,$ *r2 h CEO*$) \mapsto$ *John*

5.3 Results

Table 1 shows the result of the SP models on each of the datasets. Each of the datasets used the same train/validation/test splits.

CRE. We report the results in the same format used in the original paper: the percentage of positive samples that were identified correctly (Acc$_+$), the percentage of negative samples that were identified correctly (Acc$_-$), and the overall weighted accuracy (Acc). Except for the token-BERT reduction, all of the reductions we used surpassed their RC and SQuAD trained models, where the SP model (BERT and ALBERT) improve by more than 5% compared to the squad models. We also observed a correlation between the amount of semantic information in the templates and the model performance: SP$_{question}$ performed better than SP$_{relation}$, which in turn outperforms SP$_{token}$.

TACRED. Both our BERT-based SP and ALBERT outperform MTB model. like in CRE, there is a clear correlation between the amount of semantic data in the template and the model accuracy. This supports our claim that even though the amount of information in the relation template is negligible compared to the amount of data the model processes during

training, it still has a major effect on performance.
SemEval. The best performing model is the QA model, which also surpasses LiTian's model. Surprisingly, the token model performs better than the relation token. We explain this anomaly by looking at the relation names in SemEval. In contrast to TACRED (and CRE) the relation names in SemEval are somewhat abstract and have lower semantic similarity to the relation instances. For example, the TACRED relation "per:parents" provides more generalization and more semantic similarity to the words that actually appear in the context compared to the SemEval relation "instrument-agency" as explained in Section 3.2.

Another difference between the datasets is the difference in accuracy gain from the move to the SP objective: CRE sees the most benefit, followed by TACRED and finally SemEval. We attribute this difference to the nature of the datasets. The span-prediction based method is specially tailored to deal with the "shallow heuristics" [Rosenman et al., 2020] that CRE was made to highlight. Such challges are in turn more prevalent in TACRED than in SemEval: SemEval does not contain the "no relation" type, and the chance of any two relations appearing in the same sentence is low.

Since we didn't have access to KEPLER (the current state-of-the-art), we used the best-pretrained model available to us—ALBERT. All of our ALBERT-based SP methods outperform the current best TACRED model (KEPLER) by 2.3% F_1, despite KEPLER using external data.

One anomaly is that on ALBERT, the QA reduction performed worse than the relation reduction and even the token reduction. We explain this by looking at the relation names in TACRED, which contain parts that add generalization over different relations, while also containing parts that have a strong semantic connection to the relations. E.g *per::age* relation having the first part supporting generalization while the latter supports the semantic connection to the relation.

6. Additional Experiments

The importance of bidirectional questions. To assess the impact of using questions in both directions, we also report the ALBERT-based SP-reduction on TACRED in which we present two questions per relation, but where both questions use e_1 as the template argument and e_2 as the answer ("Single direction" in Table 2). This model has significantly less success than the two-way model, resulting in a drop of 2.4%F_1. **Combination using OR vs. AND.** We combine the answers to the two generated questions by an "OR" operator, but the same can be done with the "AND" operator. To check this we ran our models but report the relation as "present" iff the two questions return a correct answer. The results are reported in Table 2. The AND operator greatly underperforms when compared to the OR operator with a drop in F_1 of about 10%. The reason for this degradation is that the AND operator is more focused on precision, while the OR operator is more focused on recall. Over the years a major challenge of RC system was to increase recall [She et al., 2018]: it's easier for RC system to filter unrelated samples than to generalize to new patterns.

Relation to SQuAD Training. We advocated a fully supervised training of RC models as span-prediction. How well does this compare to using existing QA models, like SQuAD, in a zero-shot setting? And can we leverage the existing knowledge in QA datasets, via

Model	P	R	F_1
SP+Pretrain (BERT,unified)	68.3	63.2	65.5
SP+Pretrain (BERT,serial)	70.1	65.1	67.5
SP$_{question,BERT}$	**71.1**	72.6	**71.8**
SP-AND$_{token,BERT}$	80.1 (63.3)	54.7 (78.4)	65.0 (70.0)
SP-AND$_{relation,BERT}$	84.4 (67.0)	44.8 (76.0)	58.5 (71.2)
SP-AND$_{question,BERT}$	83.15 (71.1)	50.0 (72.6)	62.4 (71.8)
SP$_{Single\ direction}$	75.8	65.4	70.2

Table 2: **Ablations. Top**:"Fine-tuning" the SP$_{question,BERT}$ models on TACRED after SQuAD 2.0 pre-training. The SP model trained without pre-training, significantly outperforming the pre-trained variants. **Bottom**: We show that using one-way questions (on SP$_{question,ALBERT}$) and the AND operator (SP-AND) perform worse than two-way questions and the OR operator.

pre-training? We explore these two options and conclude that while the zero-shot accuracy is impressively high, the unification of SQuAD and TACRED harms the overall accuracy.

Zero-shot SQuAD. In light of the success of SQuAD trained model on CRE (as demonstrated by Rosenman et al. [2020]), we evaluate the SQuAD 2.0 trained model performance on TACRED, using our bidirectional reduction. In this zero-shot setup, we take a SQuAD trained model (without any modifications) and apply our reduction to evaluate the test set of TACRED.

Zero-shot Results. Table 2 lists the results. Unsurprisingly, the zero shots F$_1$ score on TACRED is substantially lower than all the supervised variants. However, the recall of the zero-shot setup is substantially higher: the SQuAD 2.0 model is very permissive.

Joined training with SQuAD. We now attempt to leverage the SQuAD 2.0 data to improve our RC model. We train our SP$_{question,BERT}$ model by combining SQuAD 2.0 samples and the TACRED-SP generated questions. We do this in two ways: in the **unified** version we combine the two datasets simply by shuffling together the TACRED and SQuAD questions into a single dataset. In the **serial** version we first train on the SQuAD data and then continue training the model on TACRED data.

Joint training results. Interestingly, the additional SQuAD *substantially hurt* the SP method compared to training on only the TACRED-generated questions. This highlights that the main benefit of the SP method originate from the combination of the supervised training and the span-prediction objective, and not merely from the QA form, or from the additional semantic information that is potentially embedded in the QA models.

7. Conclusion

We argue for the use of span-prediction objectives, typically used for QA, to replace the prevalent RC architectures. Our approach reduces each RC sample to two binary span-prediction tasks. We show that This approach achieves state-of-the-art performance in supervised settings, with the moderate cost of supplying question templates that describe the relation.

Acknowledgments

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme, grant agreement No. 802774 (iEXTRACT).

References

Christoph Alt, Aleksandra Gabryszak, and Leonhard Hennig. TACRED Revisited: A Thorough Evaluation of the TACRED Relation Extraction Task. apr 2020. URL http://arxiv.org/abs/2004.14855.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. Matching the Blanks: Distributional Similarity for Relation Learning. pages 2895–2905. Association for Computational Linguistics (ACL), jun 2019. ISBN 9781950737482. doi: 10.18653/v1/p19-1279. URL http://arxiv.org/abs/1906.03158.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. oct 2018. URL http://arxiv.org/abs/1810.04805.

Xinya Du and Claire Cardie. Event Extraction by Answering (Almost) Natural Questions. apr 2020. URL http://arxiv.org/abs/2004.13625.

Luheng He, Mike Lewis, and Luke Zettlemoyer. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing*, 2015. ISBN 9781941643327.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid O. Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. SemEval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *ACL 2010 - SemEval 2010 - 5th International Workshop on Semantic Evaluation, Proceedings*, pages 33–38. Association for Computational Linguistics, 2010. ISBN 1932432701. URL http://docs.

Zhengbao Jiang, Wei Xu, Jun Araki, and Graham Neubig. Generalizing Natural Language Analysis through Span-relation Representations. nov 2019. URL http://arxiv.org/abs/1911.03822.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. SpanBERT: Improving Pre-training by Representing and Predicting Spans. *Transactions of the Association for Computational Linguistics*, 8:64–77, jul 2020. ISSN 2307-387X. doi: 10.1162/tacl_a_00300. URL http://arxiv.org/abs/1907.10529.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. sep 2019. URL http://arxiv.org/abs/1909.11942.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. In *CoNLL 2017 - 21st Conference on Computational Natural Language Learning, Proceedings*, pages 333–342. Association for Computational Linguistics (ACL), jun 2017. ISBN 9781945626548. doi: 10.18653/v1/k17-1034. URL http://arxiv.org/abs/1706.04115.

Cheng Li and Ye Tian. Downstream Model Design of Pre-trained Language Model for Relation Extraction Task. apr 2020. URL http://arxiv.org/abs/2004.03786.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. A Unified MRC Framework for Named Entity Recognition. oct 2019a. URL http://arxiv.org/abs/1910.11476.

Xiaoya Li, Fan Yin, Zijun Sun, Xiayu Li, Arianna Yuan, Duo Chai, Mingxin Zhou, and Jiwei Li. Entity-Relation Extraction as Multi-Turn Question Answering. pages 1340–1350. Association for Computational Linguistics (ACL), may 2019b. doi: 10.18653/v1/p19-1129. URL http://arxiv.org/abs/1905.05529.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach. jul 2019. URL http://arxiv.org/abs/1907.11692.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. The Natural Language Decathlon: Multitask Learning as Question Answering. jun 2018. URL http://arxiv.org/abs/1806.08730.

Shachar Rosenman, Alon Jacovi, and Yoav Goldberg. Exposing Shallow Heuristics of Relation Extraction Models with Challenge Data. oct 2020. URL http://arxiv.org/abs/2010.03656.

Heng She, Bin Wu, Bai Wang, and Renjun Chi. Distant Supervision for Relation Extraction with Hierarchical Attention and Entity Descriptions. In *Proceedings of the International Joint Conference on Neural Networks*, volume 2018-July, 2018. ISBN 9781509060146. doi: 10.1109/IJCNN.2018.8489631. URL www.aaai.org.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhiyuan Liu, Juanzi Li, and Jian Tang. KEPLER: A Unified Model for Knowledge Embedding and Pre-trained Language Representation. nov 2019. URL http://arxiv.org/abs/1911.06136.

Zhepei Wei, Jianlin Su, Yue Wang, Yuan Tian, and Yi Chang. A Novel Cascade Binary Tagging Framework for Relational Triple Extraction. Technical report, 2019. URL https://github.com/weizhepei/CasRel.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. HuggingFace's Transformers: State-of-the-art Natural Language Processing. oct 2019. URL http://arxiv.org/abs/1910.03771.

Wei Wu, Fei Wang, Arianna Yuan, Fei Wu, and Jiwei Li. Coreference Resolution as Query-based Span Prediction. nov 2019. URL http://arxiv.org/abs/1911.01746.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D Manning. Position-aware attention and supervised data improve slot filling. In *EMNLP 2017 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, pages 35–45, 2017. ISBN 9781945626838. doi: 10.18653/v1/d17-1004.