Zero-Shot Relation Extraction via Reading Comprehension

Omer Levy[†]   **Minjoon Seo**[†]   **Eunsol Choi**[†]   **Luke Zettlemoyer**[†‡]

[†]Allen School of Computer Science and Engineering, Univ. of Washington, Seattle WA
[‡]Allen Institute for Artificial Intelligence, Seattle WA
{omerlevy,minjoon,eunsol,lsz}@cs.washington.edu

Abstract

We show that relation extraction can be reduced to answering simple reading comprehension questions, by associating one or more natural-language questions with each relation slot. This reduction has several advantages: we can (1) learn relation-extraction models by extending recent neural reading-comprehension techniques, (2) build very large training sets for those models by combining relation-specific crowd-sourced questions with distant supervision, and even (3) do zero-shot learning by extracting new relation types that are only specified at test-time, for which we have no labeled training examples. Experiments on a Wikipedia slot-filling task demonstrate that the approach can generalize to new questions for known relation types with high accuracy, and that zero-shot generalization to unseen relation types is possible, at lower accuracy levels, setting the bar for future work on this task.

1 Introduction

Relation extraction systems populate knowledge bases with facts from an unstructured text corpus. When the type of facts (relations) are predefined, one can use crowdsourcing (Liu et al., 2016) or distant supervision (Hoffmann et al., 2011) to collect examples and train an extraction model for each relation type. However, these approaches are incapable of extracting relations that were *not* specified in advance and observed during training. In this paper, we propose an alternative approach for relation extraction, which can potentially extract facts of new types that were neither specified nor observed a priori.

Relation	Question Template
$educated\_at(x,y)$	Where did x graduate from?
	In which university did x study?
	What is x's alma mater?
$occupation(x,y)$	What did x do for a living?
	What is x's job?
	What is the profession of x?
$spouse(x,y)$	Who is x's spouse?
	Who did x marry?
	Who is x married to?

Figure 1: Common knowledge-base relations defined by natural-language question templates.

We show that it is possible to reduce relation extraction to the problem of answering simple reading comprehension questions. We map each relation type $R(x,y)$ to at least one parametrized natural-language question q_x whose answer is y. For example, the relation $educated\_at(x,y)$ can be mapped to "Where did x study?" and "Which university did x graduate from?". Given a particular entity x ("Turing") and a text that mentions x ("Turing obtained his PhD from Princeton"), a non-null answer to any of these questions ("Princeton") asserts the fact and also fills the slot y. Figure 1 illustrates a few more examples.

This reduction enables new ways of framing the learning problem. In particular, it allows us to perform *zero-shot learning*: define new relations "on the fly", *after* the model has already been trained. More specifically, the zero-shot scenario assumes access to labeled data for N relation types. This data is used to train a reading comprehension model through our reduction. However, at test time, we are asked about a previously unseen relation type R_{N+1}. Rather than providing labeled data for the new relation, we simply list questions that define the relation's slot values. Assuming we learned a good reading comprehension model, the correct values should be extracted.

Our zero-shot setup includes innovations both

in data and models. We use distant supervision for a relatively large number of relations (120) from Wikidata (Vrandečić, 2012), which are easily gathered in practice via the WikiReading dataset (Hewlett et al., 2016). We also introduce a crowdsourcing approach for gathering and verifying the questions for each relation. This process produced about 10 questions per relation on average, yielding a dataset of over 30,000,000 question-sentence-answer examples in total. Because questions are paired with relation types, not instances, this overall procedure has very modest costs.

The key modeling challenge is that most existing reading-comprehension problem formulations assume the answer to the question is always present in the given text. However, for relation extraction, this premise does not hold, and the model needs to reliably determine when a question is not answerable. We show that a recent state-of-the-art neural approach for reading comprehension (Seo et al., 2016) can be directly extended to model answerability and trained on our new dataset. This modeling approach is another advantage of our reduction: as machine reading models improve with time, so should our ability to extract relations.

Experiments demonstrate that our approach generalizes to new paraphrases of questions from the training set, while incurring only a minor loss in performance (4% relative F1 reduction). Furthermore, translating relation extraction to the realm of reading comprehension allows us to extract a significant portion of previously unseen relations, from virtually zero to an F1 of 41%. Our analysis suggests that our model is able to generalize to these cases by learning typing information that occurs across many relations (e.g. the answer to "Where" is a location), as well as detecting relation paraphrases to a certain extent. We also find that there are many feasible cases that our model does not quite master, providing an interesting challenge for future work.

2 Related Work

We are interested in a particularly harsh zero-shot learning scenario: given labeled examples for N relation types during training, extract relations of a new type R_{N+1} at test time. The only information we have about R_{N+1} are parametrized questions.

This setting differs from prior art in relation extraction. Bronstein et al. (2015) explore a similar zero-shot setting for event-trigger identification, in which R_{N+1} is specified by a set of trigger words at test time. They generalize by measuring the similarity between potential triggers and the given seed set using unsupervised methods. We focus instead on slot filling, where questions are more suitable descriptions than trigger words.

Open information extraction (open IE) (Banko et al., 2007) is a schemaless approach for extracting facts from text. While open IE systems need no relation-specific training data, they often treat different phrasings as different relations. In this work, we hope to extract a canonical slot value independent of how the original text is phrased.

Universal schema (Riedel et al., 2013) represents open IE extractions and knowledge-base facts in a single matrix, whose rows are entity pairs and columns are relations. The redundant schema (each knowledge-base relation may overlap with multiple natural-language relations) enables knowledge-base population via matrix completion techniques. Verga et al. (2017) predict facts for entity pairs that were not observed in the original matrix; this is equivalent to extracting seen relation types with unseen entities (see Section 6.1). Rocktäschel et al. (2015) and Demeester et al. (2016) use inference rules to predict hidden knowledge-base relations from observed natural-language relations. This setting is akin to generalizing across different manifestations of the same relation (see Section 6.2) since a natural-language description of each target relation appears in the training data. Moreover, the information about the unseen relations is a set of explicit inference rules, as opposed to implicit natural-language questions.

Our zero-shot scenario, in which no manifestation of the test relation is observed during training, is substantially more challenging (see Section 6.3). In universal-schema terminology, we add a new empty column (the target knowledge-base relation), plus a few new columns with a single entry each (reflecting the textual relations in the sentence). These columns share no entities with existing columns, making the rest of the matrix irrelevant. To fill the empty column from the others, we match their descriptions. Toutanova et al. (2015) proposed a similar approach that decomposes natural-language relations and computes their similarity in a universal schema setting; however, they did not extend their method to knowledge-base relations, nor did they attempt to recover out-of-schema relations as we do.

3 Approach

We consider the slot-filling challenge in relation extraction, in which we are given a knowledge-base relation R, an entity e, and a sentence s. For example, consider the relation *occupation*, the entity "Steve Jobs", and the sentence "Steve Jobs was an American businessman, inventor, and industrial designer". Our goal is to find a set of text spans A in s for which $R(e, a)$ holds for each $a \in A$. In our example, $A =$ {businessman, inventor, industrial designer}. The empty set is also a valid answer ($A = \emptyset$) when s does not contain any phrase that satisfies $R(e, ?)$. We observe that given a natural-language question q that expresses $R(e, ?)$ (e.g. "What did Steve Jobs do for a living?"), solving the reading comprehension problem of answering q from s is equivalent to solving the slot-filling challenge.

The challenge now becomes one of *querification*: translating $R(e, ?)$ into q. Rather than querify $R(e, ?)$ for every entity e, we propose a method of querifying the relation R. We treat e as a variable x, querify the parametrized query $R(x, ?)$ (e.g. *occupation*$(x, ?)$) as a question template q_x ("What did x do for a living?"), and then instantiate this template with the relevant entities, creating a tailored natural-language question for each entity e ("What did *Steve Jobs* do for a living?"). This process, *schema querification*, is by an order of magnitude more efficient than querifying individual instances because annotating a relation type automatically annotates all of its instances.

Applying schema querification to N relations from a pre-existing relation-extraction dataset converts it into a reading-comprehension dataset. We then use this dataset to train a reading-comprehension model, which given a sentence s and a question q returns a set of text spans A within s that answer q (to the best of its ability).

In the zero-shot scenario, we are given a new relation $R_{N+1}(x, y)$ at test-time, which was neither specified nor observed beforehand. For example, the *deciphered*(x, y) relation, as in "Turing and colleagues came up with a method for efficiently deciphering the Enigma", is too domain-specific to exist in common knowledge-bases. We then querify $R_{N+1}(x, y)$ into q_x ("Which code did x break?") or q_y ("Who cracked y?"), and run our reading-comprehension model for each sentence in the document(s) of interest, while instantiating the question template with different entities

that might participate in this relation.[1] Each time the model returns a non-null answer a for a given question q_e, it extracts the relation $R_{N+1}(e, a)$.

Ultimately, all we need to do for a new relation is define our information need in the form of a question.[2] Our approach provides a natural-language API for application developers who are interested in incorporating a relation-extraction component in their programs; no linguistic knowledge or pre-defined schema is needed. To implement our approach, we require two components: training data and a reading-comprehension model. In Section 4, we construct a large relation-extraction dataset and querify it using an efficient crowdsourcing procedure. We then adapt an existing state-of-the-art reading-comprehension model to suit our problem formulation (Section 5).

4 Dataset

To collect reading-comprehension examples as in Figure 2, we first gather labeled examples for the task of relation-slot filling. Slot-filling examples are similar to reading-comprehension examples, but contain a knowledge-base query $R(e, ?)$ instead of a natural-language question; e.g. *spouse*(Angela Merkel, ?) instead of "Who is Angela Merkel married to?". We collect many slot-filling examples via distant supervision, and then convert their queries into natural language.

Slot-Filling Data We use the WikiReading dataset (Hewlett et al., 2016) to collect labeled slot-filling examples. WikiReading was collected by aligning each Wikidata (Vrandečić, 2012) relation $R(e, a)$ with the corresponding Wikipedia article D for the entity e, under the reasonable assumption that the relation can be derived from the article's text. Each instance in this dataset contains a relation R, an entity e, a document D, and an answer a. We used distant supervision to select the specific sentences in which each $R(e, a)$ manifests. Specifically, we took the first sentence s in D to contain both e and a. We then grouped instances by R, e, and s to merge all the answers for $R(e, ?)$ given s into one answer set A.

---

[1] This can be implemented efficiently by constraining potential entities with existing facts in the knowledge base. For example, any entity x that satisfies *occupation*$(x, cryptographer)$ or any entity y for which *subclass_of*$(y, cipher)$ holds. We leave the exact implementation details of such a system for future work.

[2] While we use questions, one can also use sentences with slots (clozes) to capture an almost identical notion.

Relation	Question	Sentence & Answers
educated_at	What is **Albert Einstein**'s alma mater?	**Albert Einstein** was awarded a PhD by the **University of Zürich**, with his dissertation titled...
occupation	What did **Steve Jobs** do for a living?	**Steve Jobs** was an American **businessman**, **inventor**, and **industrial designer**.
spouse	Who is **Angela Merkel** married to?	**Angela Merkel**'s second and current husband is quantum chemist and professor **Joachim Sauer**, who has largely...

Figure 2: Examples from our reading-comprehension dataset. Each instance contains a relation R, a question q, a sentence s, and an answer set A. The question explicitly mentions an entity e, which also appears in s. For brevity, answers are underlined instead of being displayed in a separate column.

> (1) The wine is produced in the **X** region of **France**.
> (2) **X**, the capital of **Mexico**, is the most populous city in North America.
> (3) **X** is an unincorporated and organized territory of **the United States**.
> (4) The **X** mountain range stretches across **the United States** and **Canada**.

Figure 3: An example of the annotator's input when querifying the $country(x, ?)$ relation. The annotator is required to ask a question about x whose answer is, for each sentence, the underlined spans.

Schema Querification Crowdsourcing querification at the schema level is not straightforward, because the task has to encourage workers to (a) figure out the relation's semantics (b) be lexically-creative when asking questions. We therefore apply a combination of crowdsourcing tactics over two Mechanical Turk annotation phases: collection and verification.

For each relation R, we present the annotator with 4 example sentences, where the entity e in each sentence s is masked by the variable x. In addition, we underline the extractable answers $a \in A$ that appear in s (see Figure 3). The annotator must then come up with a question about x whose answer, given each sentence s, is the underlined span within that sentence. For example, "In which country is x?" captures the exact set of answers for each sentence in Figure 3. Asking a more general question, such as "Where is x?" might return false positives ("North America" in sentence 2).

Each worker produced 3 different question templates for each example set. For each relation, we sampled 3 different example sets, and hired 3 different annotators for each set. We ran one instance of this annotation phase where the workers were also given, in addition to the example set, the name of the relation (e.g. *country*), and another instance where it was hidden. Out of a potential 54 question templates, 40 were unique on average.

In the verification phase, we measure the question templates' quality by sampling additional sentences and instantiating each question template with the example entity e. Annotators are then asked to answer the question from the sentence s, or mark it as unanswerable; if the annotators' an-

swers match A, the question template is valid. We discarded the templates that were not answered correctly in the majority of the examples (6/10).[3]

Overall, we applied schema querification to 178 relations that had at least 100 examples each (accounting for 99.77% of the data), costing roughly $1,250. After the verification phase, we were left with 1,192 high-quality question templates spanning 120 relations.[4] We then join these templates with our slot-filling dataset along relations, instantiating each template q_x with its matching entities. This process yields a reading-comprehension dataset of over 30,000,000 examples, where each instance contains the original relation R (unobserved by the machine), a question q, a sentence s, and the set of answers A (see Figure 2).

Negative Examples To support relation extraction, our dataset deviates from recent reading comprehension formulations (Hermann et al., 2015; Rajpurkar et al., 2016), and introduces negative examples – question-sentence pairs that have no answers ($A = \emptyset$). Following the methodology of InfoboxQA (Morales et al., 2016), we generate negative examples by matching (for the same entity e) a question q that pertains to one relation with a sentence s that expresses another relation. We also assert that the sentence does not contain the answer to q. For instance, we match "Who

---

[3]We used this relatively lenient measure because many annotators selected the correct answer, but with a slightly incorrect span; e.g. "American businessman" instead of "businessman". We therefore used token-overlap F1 as a secondary filter, requiring an average score of at least 0.75.

[4]58 relations had zero questions after verification due to noisy distant supervision and little annotator quality control.

is Angela Merkel married to?" with a sentence about her occupation: "Angela Merkel is a German politician who is currently the Chancellor of Germany." This process generated over 2 million negative examples. While this is a relatively naive method of generating negative examples, our analysis shows that about a third of negative examples contain good distractors (see Section 7).

Discussion Some recent QA datasets were collected by expressing knowledge-base assertions in natural language. The Simple QA dataset (Bordes et al., 2015) was created by annotating questions about individual Freebase facts (e.g. $educated\_at(Turing, Princeton)$), collecting roughly 100,000 natural-language questions to support QA against a knowledge graph. Morales et al. (2016) used a similar process to collect questions from Wikipedia infoboxes, yielding the 15,000-example InfoboxQA dataset. For the task of identifying predicate-argument structures, QA-SRL (He et al., 2015) was proposed as an open schema for semantic roles, in which the relation between an argument and a predicate is expressed as a natural-language question containing the predicate ("Where was someone educated?") whose answer is the argument ("Princeton"). The authors collected about 19,000 question-answer pairs from 3,200 sentences.

In these efforts, the costs scale linearly in the number of instances, requiring significant investments for large datasets. In contrast, schema querification can generate an enormous amount of data for a fraction of the cost by labeling at the relation level; as evidence, we were able to generate a dataset 300 times larger than Simple QA. To the best of our knowledge, this is the first robust method for collecting a question-answering dataset by crowd-annotating at the schema level.

5   Model

Given a sentence s and a question q, our algorithm either returns an answer span[5] a within s, or indicates that there is no answer.

The task of obtaining answer spans to natural-language questions has been recently studied on the SQuAD dataset (Rajpurkar et al., 2016; Xiong et al., 2016; Lee et al., 2016; Wang et al., 2016). In SQuAD, every question is answerable from the

---

[5]While our problem definition allows for multiple answer spans per question, our algorithm assumes a single span; in practice, less than 5% of our data has multiple answers.

text, which is why these models assume that there exists a correct answer span. Therefore, we modify an existing model in a way that allows it to decide whether an answer exists. We first give a high-level description of the original model, and then describe our modification.

We start from the BiDAF model (Seo et al., 2016), whose input is two sequences of words: a sentence s and a question q. The model predicts the start and end positions $\mathbf{y}^{start}, \mathbf{y}^{end}$ of the answer span in s. BiDAF uses recurrent neural networks to encode contextual information within s and q alongside an attention mechanism to align parts of q with s and vice-versa.

The outputs of the BiDAF model are the confidence scores of $\mathbf{y}^{start}$ and $\mathbf{y}^{end}$, for each potential start and end. We denote these scores as $\mathbf{z}^{start}, \mathbf{z}^{end} \in \mathbb{R}^N$, where N is the number of words in the sentence s. In other words, $\mathbf{z}_i^{start}$ indicates how likely the answer is to start at position i of the sentence (the higher the more likely); similarly, $\mathbf{z}_i^{end}$ indicates how likely the answer is to end at that index. Assuming the answer exists, we can transform these confidence scores into pseudo-probability distributions $\mathbf{p}^{start}, \mathbf{p}^{end}$ via softmax. The probability of each i-to-j-span of the context can therefore be defined by:

$$P(a = s_{i...j}) = \mathbf{p}_i^{start}\mathbf{p}_j^{end} \qquad (1)$$

where $\mathbf{p}_i$ indicates the i-th element of the vector $\mathbf{p}_i$, i.e. the probability of the answer starting at i. Seo et al. (2016) obtain the span with the highest probability during post-processing.

To allow the model to signal that there is no answer, we concatenate a trainable bias b to the end of both confidences score vectors $\mathbf{z}^{start}, \mathbf{z}^{end}$. The new score vectors $\tilde{\mathbf{z}}^{start}, \tilde{\mathbf{z}}^{end} \in \mathbb{R}^{N+1}$ are defined as $\tilde{\mathbf{z}}^{start} = [\mathbf{z}^{start}; b]$ and similarly for $\tilde{\mathbf{z}}^{end}$, where $[;]$ indicates row-wise concatenation. Hence, the last elements of $\tilde{\mathbf{z}}^{start}$ and $\tilde{\mathbf{z}}^{end}$ indicate the model's confidence that the answer has no start or end, respectively. We apply softmax to these augmented vectors to obtain pseudo-probability distributions, $\tilde{\mathbf{p}}^{start}, \tilde{\mathbf{p}}^{end}$. This means that the probability the model assigns to a null answer is:

$$P(a = \emptyset) = \tilde{\mathbf{p}}_{N+1}^{start}\tilde{\mathbf{p}}_{N+1}^{end}. \qquad (2)$$

If $P(a = \emptyset)$ is higher than the probability of the best span, $\arg\max_{i,j \le N} P(a = s_{i...j})$, then the model deems that the question cannot be answered from the sentence. Conceptually, adding the bias

enables the model to be sensitive to the absolute values of the raw confidence scores $\mathbf{z}^{start}, \mathbf{z}^{end}$. We are essentially setting and learning a threshold b that decides whether the model is sufficiently confident of the best candidate answer span.

While this threshold provides us with a dynamic per-example decision of whether the instance is answerable, we can also set a global confidence threshold p_{min}; if the best answer's confidence is below that threshold, we infer that there is no answer. In Section 6.3 we use this global threshold to get a broader picture of the model's performance.

6 Experiments

To understand how well our method can generalize to unseen data, we design experiments for unseen entities (Section 6.1), unseen question templates (Section 6.2), and unseen relations (Section 6.3).

Evaluation Metrics Each instance is evaluated by comparing the tokens in the labeled answer set with those of the predicted span.[6] Precision is the true positive count divided by the number of times the system returned a non-null answer. Recall is the true positive count divided by the number of instances that have an answer.

Hyperparameters In our experiments, we initialized word embeddings with GloVe (Pennington et al., 2014), and did not fine-tune them. The typical training set was an order of 1 million examples, for which 3 epochs were enough for convergence. All training sets had a ratio of 1:1 positive and negative examples, which was chosen to match the test sets' ratio.

Comparison Systems We experiment with several variants of our model. In *KB Relation*, we feed our model a relation indicator (e.g. R_{17}) instead of a question. We expect this variant to generalize reasonably well to unseen entities, but fail on unseen relations. The second variant (*NL Relation*) uses the relation's name (as a natural-language expression) instead of a question (e.g. *educated_at* as "educated at"). We also consider a weakened version of our querification approach (*Single Template*) where, during training, only one question template per relation is observed. The full variant of our model, *Multiple Templates*, is

---

[6] We ignore word order, case, punctuation, and articles ("a", "an", "the"). We also ignore "and", which often appears when a single span captures multiple correct answers (e.g. "United States and Canada").

trained on a more diverse set of questions. We expect this variant to have significantly better paraphrasing abilities than *Single Template*.

We also evaluate how asking about the same relation in multiple ways improves performance (*Question Ensemble*). We create an ensemble by sampling 3 questions per test instance and predicting the answer for each. We then choose the answer with the highest sum of confidence scores.

In addition to our model, we compare three other systems. The first is a random baseline that chooses a named entity in the sentence that does not appear in the question (*Random NE*). We also reimplement the *RNN Labeler* that was shown to have good results on the extractive portion of WikiReading (Hewlett et al., 2016). Lastly, we retrain an off-the-shelf relation extraction system (Miwa and Bansal, 2016), which has shown promising results on a number of benchmarks. This system (and many like it) represents relations as indicators, and cannot extract unseen relations.

6.1 Unseen Entities

We show that our reading-comprehension approach works well in a typical relation-extraction setting by testing it on unseen entities and texts.

Setup We partitioned our dataset along entities in the question, and randomly clustered each entity into one of three groups: train, dev, or test. For instance, Alan Turing examples appear only in training, while Steve Jobs examples are exclusive to test. We then sampled 1,000,000 examples for train, 1,000 for dev, and 10,000 for test. This partition also ensures that the sentences at test time are different from those in train, since the sentences are gathered from each entity's Wikipedia article.

Results Table 1 shows that our model generalizes well to new entities and texts, with little variance in performance between *KB Relation*, *NL Relation*, *Multiple Templates*, and *Question Ensemble*. *Single Template* performs significantly worse than these variants; we conjecture that simpler relation descriptions (*KB Relation* & *NL Relation*) allow for easier parameter tying across different examples, whereas learning from multiple questions allows the model to acquire important paraphrases. All variants of our model outperform off-the-shelf relation extraction systems (*RNN Labeler* and *Miwa & Bansal*) in this setting, demonstrating that reducing relation extraction to reading

	Precision	Recall	F1
Random NE	11.17%	22.14%	14.85%
RNN Labeler	62.55%	62.25%	62.40%
Miwa & Bansal	96.07%	58.70%	72.87%
KB Relation	89.08%	91.54%	90.29%
NL Relation	88.23%	91.02%	89.60%
Single Template	77.92%	73.88%	75.84%
Multiple Templates	87.66%	91.32%	89.44%
Question Ensemble	88.08%	91.60%	89.80%

Table 1: Performance on unseen entities.

	Precision	Recall	F1
Seen	86.73%	86.54%	86.63%
Unseen	84.37%	81.88%	83.10%

Table 2: Performance on seen/unseen questions.

comprehension is indeed a viable approach for our Wikipedia slot-filling task.

An analysis of 50 examples that *Multiple Templates* mispredicted shows that 36% of errors can be attributed to annotation errors (chiefly missing entries in Wikidata), and an additional 42% result from inaccurate span selection (e.g. "8 February 1985" instead of "1985"), for which our model is fully penalized. In total, only 18% of our sample were pure system errors, suggesting that our model is very close to the performance ceiling of this setting (slightly above 90% F1).

6.2 Unseen Question Templates

We test our method's ability to generalize to new descriptions of the same relation, by holding out a question template for each relation during training.

Setup We created 10 folds of train/dev/test samples of the data, in which one question template for each relation was held out for the test set, and another for the development set. For instance, "What did x do for a living?" may appear only in the training set, while "What is x's job?" is exclusive to the test set. Each split was stratified by sampling N examples per question template ($N = 1000, 10, 50$ for train, dev, test, respectively). This process created 10 training sets of 966,000 examples with matching development and test sets of 940 and 4,700 examples each.

We trained and tested *Multiple Templates* on each one of the folds, yielding performance on unseen templates. We then replicated the existing test sets and replaced the unseen question templates with templates from the training set, yielding performance on seen templates. Revisiting our example, we convert test-set occurrences of "What is x's job?" to "What did x do for a living?".

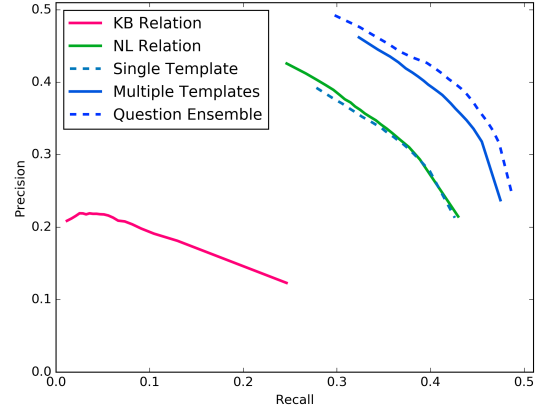|	Precision	Recall	F1
Random NE	9.25%	18.06%	12.23%
RNN Labeler	13.28%	5.69%	7.97%
Miwa & Bansal	100.00%	0.00%	0.00%
KB Relation	19.32%	2.54%	4.32%
NL Relation	40.50%	28.56%	33.40%
Single Template	37.18%	31.24%	33.90%
Multiple Templates	43.61%	36.45%	39.61%
Question Ensemble	45.85%	37.44%	41.11%

Table 3: Performance on unseen relations.

Figure 4: Precision/Recall for unseen relations.

Results Table 2 shows that our approach is able to generalize to unseen question templates. Our system's performance on unseen questions is nearly as strong as for previously observed templates (losing roughly 3.5 points in F1).

6.3 Unseen Relations

We examine a pure zero-shot setting, where test-time relations are unobserved during training.

Setup We created 10 folds of train/dev/test samples, partitioned along relations: 84 relations for train, 12 dev, and 24 test. For example, when $educated\_at$ is allocated to test, no $educated\_at$ examples appear in train. Using stratified sampling of relations, we created 10 training sets of 840,000 examples each with matching dev and test sets of 600 and 12,000 examples per fold.

Results Table 3 shows each system's performance; Figure 4 extends these results for variants of our model by applying a global threshold on the answers' confidence scores to generate precision/recall curves (see Section 5). As expected, representing knowledge-base relations as indicators (*KB Relation* and *Miwa & Bansal*) is insufficient in a zero-shot setting; they must be interpreted as natural-language expressions to allow for

Verbatim	Relation	András Dombai **plays for** what team? András Dombai... ...currently **plays** as a goalkeeper **for** *FC Tatabánya*.
	Type	Which **airport** is most closely associated with Royal Jordanian? Royal Jordanian Airlines... ...from its main base at *Queen Alia International **Airport***...
Global	Relation	Who was responsible for **directing** Les petites fugues? Les petites fugues is a 1979 Swiss comedy film **directed by** *Yves Yersin*.
	Type	**When** was The Snow Hawk released? The Snow Hawk is a *1925* film...
Specific	Relation	Who **started** Fürstenberg China? The Fürstenberg China Factory **was founded**... ...**by** *Johann Georg von Langen*...
	Type	What **voice type** does Étienne Lainez have? Étienne Lainez... ...was a French operatic ***tenor***...

Figure 5: The different types of discriminating cues we observed among positive examples.

some generalization. The difference between using a single question template (*Single Template*) and the relation's name (*NL Relation*) appears to be minor. However, training on a variety of question templates (*Multiple Templates*) substantially increases performance. We conjecture that multiple phrasings of the same relation allows our model to learn answer-type paraphrases that occur across many relations (see Section 7). There is also some advantage to having multiple questions at test time (*Question Ensemble*).

7 Analysis

To understand how our method extracts unseen relations, we analyzed 100 random examples, of which 60 had answers in the sentence and 40 did not (negative examples).

For negative examples, we checked whether a distractor – an incorrect answer of the correct answer type – appears in the sentence. For example, the question "Who is John McCain married to?" does not have an answer in "John McCain chose Sarah Palin as his running mate", but "Sarah Palin" is of the correct answer type. We noticed that 14 negative examples (35%) contain distractors. When pairing these examples with the results from the unseen relations experiment in Section 6.3, we found that our method answered 2/14 of the distractor examples incorrectly, compared to only 1/26 of the easier examples. It appears that while most of the negative examples are easy, a significant portion of them are not trivial.

For positive examples, we observed that some instances can be solved by matching the relation in the sentence to that in the question, while others rely more on the answer's type. Moreover, we notice that each cue can be further categorized according to the type of information needed to detect it: (1) when part of the question appears verba-

	Relation	Type
Verbatim	12%	5%
Global	8%	25%
Specific	22%	28%

Table 4: The distribution of cues by type, based on a sample of 60.

	Relation	Type
Verbatim	*43%*	*33%*
Global	*60%*	73%
Specific	46%	18%

Table 5: Our method's accuracy on subsets of examples pertaining to different cue types. Results in *italics* are based on a sample of less than 10.

tim in the text, (2) when the phrasing in the text deviates from the question in a way that is typical of other relations as well (e.g. syntactic variability), (3) when the phrasing in the text deviates from the question in a way that is unique to this relation (e.g. lexical variability). We name these categories *verbatim*, *global*, and *specific*, respectively. Figure 5 illustrates all the different types of cues we discuss in our analysis.

We selected the most important cue for solving each instance. If there were two important cues, each one was counted as half. Table 4 shows their distribution. Type cues appear to be somewhat more dominant than relation cues (58% vs. 42%). Half of the cues are relation-specific, whereas global cues account for one third of the cases and verbatim cues for one sixth. This is an encouraging result, because we can potentially learn to accurately recognize verbatim and global cues from other relations. However, our method was only able to exploit these cues partially.

We paired these examples with the results from the unseen relations experiment in Section 6.3 to see how well our method performs in each category. Table 5 shows the results for the *Multiple*

Templates setting. On one hand, the model appears agnostic to whether the relation cue is verbatim, global, or specific, and is able to correctly answer these instances with similar accuracy (there is no clear trend due to the small sample size). For examples that rely on typing information, the trend is much clearer; our model is much better at detecting global type cues than specific ones.

Based on these observations, we think that the primary sources of our model's ability to generalize to new relations are: *global type detection*, which is acquired from training on many different relations, and *relation paraphrase detection* (of all types), which probably relies on its pre-trained word embeddings.

8   Conclusion

We showed that relation extraction can be reduced to a reading comprehension problem, allowing us to generalize to unseen relations that are defined on-the-fly in natural language. However, the problem of zero-shot relation extraction is far from solved, and poses an interesting challenge to both the information extraction and machine reading communities. As research into machine reading progresses, we may find that more tasks can benefit from a similar approach. To support future work in this avenue, we make our code and data publicly available.[7]

Acknowledgements

References

Michele Banko, Michael J. Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, IJCAI'07, pages 2670–2676. http://dl.acm.org/citation.cfm?id=1625275.1625705.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075* .

Ofer Bronstein, Ido Dagan, Qi Li, Heng Ji, and Anette Frank. 2015. Seed-based event trigger labeling: How far can event descriptions get us? In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Association for Computational Linguistics, Beijing, China, pages 372–376. http://www.aclweb.org/anthology/P15-2061.

Thomas Demeester, Tim Rocktäschel, and Sebastian Riedel. 2016. Lifted rule injection for relation embeddings. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Austin, Texas, pages 1389–1399. https://aclweb.org/anthology/D16-1146.

Luheng He, Mike Lewis, and Luke Zettlemoyer. 2015. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 643–653. http://aclweb.org/anthology/D15-1076.

Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*. http://arxiv.org/abs/1506.03340.

Daniel Hewlett, Alexandre Lacoste, Llion Jones, Illia Polosukhin, Andrew Fandrianto, Jay Han, Matthew Kelcey, and David Berthelot. 2016. Wikireading: A novel large-scale language understanding task over wikipedia. In *Proceedings of the Conference of the Association for Computational Linguistics*.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, pages 541–550.

Kenton Lee, Tom Kwiatkowski, Ankur Parikh, and Dipanjan Das. 2016. Learning recurrent span representations for extractive question answering. *arXiv preprint arXiv:1611.01436* .

Angli Liu, Stephen Soderland, Jonathan Bragg, Christopher H. Lin, Xiao Ling, and Daniel S. Weld. 2016. Effective crowd annotation for relation extraction. In *Proceedings of the 2016 Conference*

---

[7] http://nlp.cs.washington.edu/zeroshot

of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, San Diego, California, pages 897–906. http://www.aclweb.org/anthology/N16-1104.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, Berlin, Germany, pages 1105–1116. http://www.aclweb.org/anthology/P16-1105.

Alvaro Morales, Varot Premtoon, Cordelia Avery, Sue Felshin, and Boris Katz. 2016. Learning to answer questions from wikipedia infoboxes. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Austin, Texas, pages 1930–1935. https://aclweb.org/anthology/D16-1199.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics, Doha, Qatar, pages 1532–1543. http://www.aclweb.org/anthology/D14-1162.

P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In Proceedings of the Conference of the Empirical Methods in Natural Language Processing.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, Atlanta, Georgia, pages 74–84. http://www.aclweb.org/anthology/N13-1008.

Tim Rocktäschel, Sameer Singh, and Sebastian Riedel. 2015. Injecting logical background knowledge into embeddings for relation extraction. In Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, Denver, Colorado, pages 1119–1129. http://www.aclweb.org/anthology/N15-1118.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. arXiv preprint arXiv:1611.01603 .

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Lisbon, Portugal, pages 1499–1509. http://aclweb.org/anthology/D15-1174.

Patrick Verga, Arvind Neelakantan, and Andrew McCallum. 2017. Generalizing to unseen entities and entity pairs with row-less universal schema. In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers. Association for Computational Linguistics, Valencia, Spain, pages 613–622. http://www.aclweb.org/anthology/E17-1058.

Denny Vrandečić. 2012. Wikidata: A new platform for collaborative data collection. In Proceedings of the 21st international conference companion on World Wide Web. ACM, pages 1063–1064.

Zhiguo Wang, Haitao Mi, Wael Hamza, and Radu Florian. 2016. Multi-perspective context matching for machine comprehension. arXiv preprint arXiv:1612.04211 .

Caiming Xiong, Victor Zhong, and Richard Socher. 2016. Dynamic coattention networks for question answering. arXiv preprint arXiv:1611.01604 .