What do Entity-Centric Models Learn? Insights from Entity Linking in Multi-Party Dialogue

Laura Aina* **Carina Silberer*** **Ionut-Teodor Sorodoc***
Matthijs Westera* **Gemma Boleda**

Universitat Pompeu Fabra
Barcelona, Spain
`{firstname.lastname}@upf.edu`

Abstract

Humans use language to refer to entities in the external world. Motivated by this, in recent years several models that incorporate a bias towards learning entity representations have been proposed. Such entity-centric models have shown empirical success, but we still know little about why.

In this paper we analyze the behavior of two recently proposed entity-centric models in a referential task, Entity Linking in Multi-party Dialogue (SemEval 2018 Task 4). We show that these models outperform the state of the art on this task, and that they do better on lower frequency entities than a counterpart model that is not entity-centric, with the same model size. We argue that making models entity-centric naturally fosters good architectural decisions. However, we also show that these models do not really build entity representations and that they make poor use of linguistic context. These negative results underscore the need for model analysis, to test whether the motivations for particular architectures are borne out in how models behave when deployed.

1 Introduction

Modeling reference to entities is arguably crucial for language understanding, as humans use language to talk about things in the world. A hypothesis in recent work on referential tasks such as co-reference resolution and entity linking (Haghighi and Klein, 2010; Clark and Manning, 2016; Henaff et al., 2017; Aina et al., 2018; Clark et al., 2018) is that encouraging models to learn and use entity representations will help them better carry out referential tasks. To illustrate, creating an entity representation with the relevant information upon reading *a woman* should make it easier to resolve a pronoun mention like *she*.[1] In the mentioned work, several models have been proposed that incorporate an explicit bias towards entity representations. Such **entity-centric** models have shown empirical success, but we still know little about what it is that they effectively learn to model. In this analysis paper, we adapt two previous entity-centric models (Henaff et al., 2017; Aina et al., 2018) for a recently proposed referential task and show that, despite their strengths, they are still very far from modeling entities.[2]

JOEY TRIBBIANI (183):
"... *see Ross, because I think you love her .*"

Ross	I	you	her
335	183	335	306

Figure 1: Character identification: example.

The task is character identification on multi-party dialogue as posed in SemEval 2018 Task 4 (Choi and Chen, 2018).[3] Models are given dialogues from the TV show *Friends* and asked to link entity mentions (nominal expressions like *I*, *she* or *the woman*) to the characters to which they refer in each case. Figure 1 shows an example, where the mentions *Ross* and *you* are linked to entity 335, mention *I* to entity 183, etc. Since the TV series revolves around a set of entities that recur over many scenes and episodes, it is a good benchmark to analyze whether entity-centric models learn and use entity representations for referential tasks.

Our contributions are three-fold: First, we adapt two previous entity-centric models and show that they do better on lower frequency entities

*denotes equal contribution.

[1]Note the analogy with traditional models in formal linguistics like Discourse Representation Theory (Kamp and Reyle, 2013).

[2]Source code for our model, the training procedure and the new dataset is published on `https://github.com/amore-upf/analysis-entity-centric-nns`.

[3]`https://competitions.codalab.org/competitions/17310`.

(a significant challenge for current data-hungry models) than a counterpart model that is not entity-centric, with the same model size. Second, through analysis we provide insights into how they achieve these improvements, and argue that making models entity-centric fosters architectural decisions that result in good inductive biases. Third, we create a dataset and task to evaluate the models' ability to encode entity information such as gender, and show that models fail at it. More generally, our paper underscores the need for the analysis of model behavior, not only through ablation studies, but also through the targeted probing of model representations (Linzen et al., 2016; Conneau et al., 2018).

2 Related Work

Modeling. Various memory architectures have been proposed that are not specifically for entity-centric models, but could in principle be employed in them (Graves et al., 2014; Sukhbaatar et al., 2015; Joulin and Mikolov, 2015; Bansal et al., 2017). The two models we base our results on (Henaff et al., 2017; Aina et al., 2018) were explicitly motivated as entity-centric. We show that our adaptations yield good results and provide a closer analysis of their behavior.

Tasks. The task of entity linking has been formalized as resolving entity mentions to referential entity entries in a knowledge repository, mostly Wikipedia (Bunescu and Paşca, 2006; Mihalcea and Csomai, 2007 and much subsequent work; for recent approaches see Francis-Landau et al., 2016; Chen et al., 2018). In the present entity linking task, only a list of entities is given, without associated encyclopedic entries, and information about the entities needs to be acquired from scratch through the task; note the analogy to how a human audience might get familiar with the TV show characters by watching it. Moreover, it addresses multi-party dialogue (as opposed to, typically, narrative text), where speaker information is crucial. A task closely related to entity linking is *coreference resolution*, i.e., predicting which portions of a text refer to the same entity (e.g., *Marie Curie* and *the scientist*). This typically requires clustering mentions that refer to the same entity (Pradhan et al., 2011). Mention clusters essentially correspond to entities, and recent work on coreference and language modeling has started exploiting an explicit notion of entity (Haghighi and Klein, 2010; Clark and Manning, 2016; Yang et al., 2017). Previous work both on entity linking and on coreference resolution (cited above, as well as Wiseman et al., 2016) often presents more complex models that incorporate e.g. hand-engineered features. In contrast, we keep our underlying model basic since we want to systematically analyze how certain architectural decisions affect performance. For the same reason we deviate from previous work to entity linking that uses a specialized coreference resolution module (e.g., Chen et al., 2017).

Analysis of Neural Network Models. Our work joins a recent strand in NLP that systematically analyzes what different neural network models learn about language (Linzen et al., 2016; Kádár et al., 2017; Conneau et al., 2018; Gulordava et al., 2018b; Nematzadeh et al., 2018, a.o.). This work, like ours, has yielded both positive and negative results: There is evidence that they learn complex linguistic phenomena of morphological and syntactic nature, like long distance agreement (Gulordava et al., 2018b; Giulianelli et al., 2018), but less evidence that they learn how language relates to situations; for instance, Nematzadeh et al. (2018) show that memory-augmented neural models fail on tasks that require keeping track of inconsistent states of the world.

3 Models

We approach character identification as a classification task, and compare a baseline LSTM (Hochreiter and Schmidhuber, 1997) with two models that enrich the LSTM with a memory module designed to learn and use entity representations. LSTMs are the workhorse for text processing, and thus a good baseline to assess the contribution of this module. The LSTM processes text of dialogue scenes one token at a time, and the output is a probability distribution over the entities (the set of entity IDs are given).

3.1 Baseline: BILSTM

The BILSTM model is depicted in Figure 2. It is a standard bidirectional LSTM (Graves et al., 2005), with the difference with most uses of LSTMs in NLP that we incorporate speaker information in addition to the linguistic content of utterances.

The model is given chunks of dialogue (see Appendix for hyperparameter settings such as the chunk size). At each time step i, one-hot vectors for token $\mathbf{t}_i$ and speaker entities $\mathbf{s}_i$ are embedded

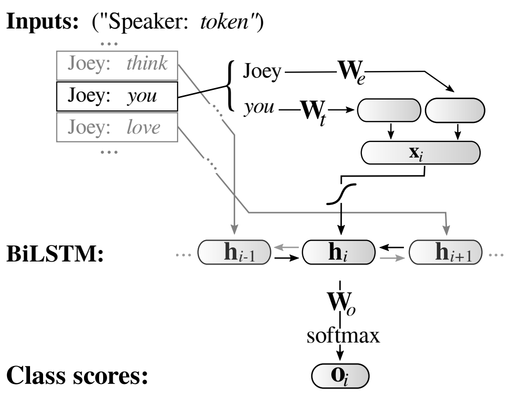

Figure 2: BILSTM applied to "...think you love..." as spoken by Joey (from Figure 1), outputting class scores for mention "you" (bias b_o not depicted).

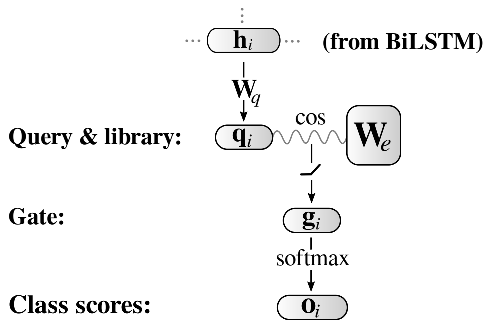

Figure 3: ENTLIB; everything before $\mathrm{h_i}$, omitted here, is the same as in Figure 2.

via two distinct matrices $\mathbf{W}_t$ and $\mathbf{W}_e$ and concatenated to form a vector $\mathbf{x}_i$ (Eq. 1, where $\|$ denotes concatenation; note that in case of multiple simultaneous speakers S_i their embeddings are summed).

$$\mathbf{x}_i = \mathbf{W}_t\, \mathbf{t}_i \,\|\, \sum_{\mathbf{s}\in S_i} \mathbf{W}_e\, \mathbf{s} \tag{1}$$

The vector $\mathbf{x}_i$ is fed through the nonlinear activation function tanh and input to a bidirectional LSTM. The hidden state $\overrightarrow{\mathbf{h}}_i$ of a *uni*directional LSTM for the i^{th} input is recursively defined as a combination of that input with the LSTM's previous hidden state $\overrightarrow{\mathbf{h}}_{i-1}$. For a *bi*directional LSTM, the hidden state $\mathbf{h}_i$ is the concatenation of the hidden states $\overrightarrow{\mathbf{h}}_i$ and $\overleftarrow{\mathbf{h}}_i$ of two unidirectional LSTMs which process the data in opposite directions (Eqs. 2-4).

$$\overrightarrow{\mathbf{h}}_i = \mathrm{LSTM}(\tanh(\mathbf{x}_i), \overrightarrow{\mathbf{h}}_{i-1}) \tag{2}$$
$$\overleftarrow{\mathbf{h}}_i = \mathrm{LSTM}(\tanh(\mathbf{x}_i), \overleftarrow{\mathbf{h}}_{i+1}) \tag{3}$$
$$\mathbf{h}_i = \overrightarrow{\mathbf{h}}_i \,\|\, \overleftarrow{\mathbf{h}}_i \tag{4}$$

For every entity mention $\mathbf{t}_i$ (i.e., every token[4] that is tagged as referring to an entity), we obtain a distribution over all entities, $\mathbf{o}_i \in [0,1]^{1\times N}$, by applying a linear transformation to its hidden state $\mathbf{h}_i$ (Eq. 5), and feeding the resulting $\mathbf{g}_i$ to a softmax classifier (Eq. 6).

$$\mathbf{g}_i = \mathbf{W}_o\, \mathbf{h}_i + \mathbf{b}_o \tag{5}$$
$$\mathbf{o}_i = \mathrm{softmax}(\mathbf{g}_i) \tag{6}$$

Eq. 5 is where the other models will diverge.

3.2 ENTLIB (Static Memory)

The ENTLIB model (Figure 3) is an adaptation of our previous work in Aina et al. (2018), which was the winner of the SemEval 2018 Task 4 competition. This model adds a simple memory module that is expected to represent entities because its vectors are tied to the output classes (accordingly, Aina et al., 2018, call this module *entity library*). We call this memory 'static', since it is updated only during training, after which it remains fixed.

Where BILSTM maps the hidden state $\mathbf{h}_i$ to class scores $\mathbf{o}_i$ with a single transformation (plus softmax), ENTLIB instead takes two steps: It first transforms $\mathbf{h}_i$ into a 'query' vector $\mathbf{q}_i$ (Eq. 7) that it will then use to query the entity library. As we will see, this mechanism helps dividing the labor between representing the context (hidden layer) and doing the prediction task (query layer).

$$\mathbf{q}_i = \mathbf{W}_q\, \mathbf{h}_i + \mathbf{b_q} \tag{7}$$

A weight matrix $\mathbf{W}_e$ is used as the entity library, which is the same as the speaker embedding in Eq. 1: the query vector $\mathbf{q}_i \in \mathbb{R}^{1\times k}$ is compared to each vector in $\mathbf{W}_e$ (cosine), and a *gate* vector $\mathbf{g}_i$ is obtained by applying the ReLU function to the cosine similarity scores (Eq. 8).[5] Thus, the query extracted from the LSTM's hidden state is used as a soft pointer over the model's representation of the entities.

$$\mathbf{g}_i = \mathrm{ReLU}(\cos(\mathbf{W}_e, \mathbf{q}_i)) \tag{8}$$

As before, a softmax over $\mathbf{g}_i$ then yields the distribution over entities (Eq. 6). So, in the ENTLIB

[4]For multi-word mentions this is done only for the last token in the mention.

[5]In Aina et al. (2018), the gate did not include the ReLU nonlinear activation function. Adding it improved results.

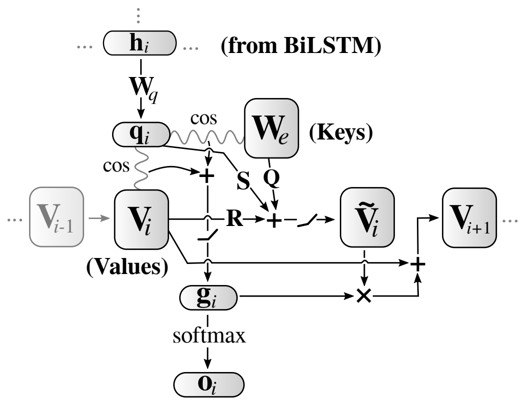

Figure 4: ENTNET; everything before $\mathrm{h_i}$, omitted here, is the same as in Figure 2.

model Eqs. 7 and 8 together take the place of Eq. 5 in the BILSTM model.

Our implementation differs from Aina et al. (2018) in one important point that we will show to be relevant to model less frequent entities (training also differs, see Section 4): The original model did not do parameter sharing between speakers and referents, but used two distinct weight matrices.

Note that the contents of the entity library in ENTLIB do not change during forward propagation of activations, but only during backpropagation of errors, i.e., during training, when the weights of $\mathbf{W}_e$ are updated. If anything, they will encode permanent properties of entities, not properties that change within a scene or between scenes or episodes, which should be useful for reference. The next model attempts to overcome this limitation.

3.3 ENTNET (Dynamic Memory)

ENTNET is an adaptation of *Recurrent Entity Networks* (Henaff et al., 2017, Figure 4) to the task. Instead of representing each entity by a single vector, as in ENTLIB, here each entity is represented jointly by a context-invariant or 'static' *key* and a context-dependent or 'dynamic' *value*. For the keys the entity embedding $\mathbf{W}_e$ is used, just like the entity library of ENTLIB. But the values $\mathbf{V}_i$ can be dynamically updated throughout a scene.

As before, an entity query $\mathbf{q}_i$ is first obtained from the BILSTM (Eq. 7). Then, ENTNET computes gate values $\mathbf{g}_i$ by estimating the query's similarity to both keys and values, as in Eq. 9 (replacing Eq. 8 of ENTLIB).[6] Output scores $\mathbf{o}_i$ are computed as in the previous models (Eq. 6).

$$\mathbf{g}_i = \mathrm{ReLU}(\cos(\mathbf{W}_e, \mathbf{q}_i) + \cos(\mathbf{V}_i, \mathbf{q}_i)) \quad (9)$$

The values $\mathbf{V}_i$ are initialized at the start of every scene ($i = 0$) as being identical to the keys ($\mathbf{V}_0 = \mathbf{W}_e$). After processing the i^{th} token, new information can be added to the values. Eq. 10 computes this new information $\tilde{\mathbf{V}}_{i,j}$, for the j^{th} entity, where $\mathbf{Q}$, $\mathbf{R}$ and $\mathbf{S}$ are learned linear transformations and PReLU denotes the parameterized rectified linear unit (He et al., 2015):

$$\tilde{\mathbf{V}}_{i,j} = \mathrm{PReLU}(\mathbf{Q}\mathbf{W}_{ej} + \mathbf{R}\mathbf{V}_{i,j} + \mathbf{S}\mathbf{q}_i) \quad (10)$$

This information $\tilde{\mathbf{V}}_{i,j}$, multiplied by the respective gate $\mathbf{g}_{i,j}$, is added to the values to be used when processing the next ($i + 1^{\text{th}}$) token (Eq. 11), and the result is normalized (Eq. 12):

$$\mathbf{V}_{i+1,j} = \mathbf{V}_j + \mathbf{g}_{i,j} * \tilde{\mathbf{V}}_{i,j} \quad (11)$$

$$\mathbf{V}_{i+1,j} = \frac{\mathbf{V}_{i+1,j}}{\|\mathbf{V}_{i+1,j}\|} \quad (12)$$

Our adaptation of the Recurrent Entity Network involves two changes. First, we use a biLSTM to process the linguistic utterance, while Henaff et al. (2017) used a simple multiplicative mask (we have natural dialogue, while their main evaluation was on bAbI, a synthetic dataset). Second, in the original model the gates were used to retrieve and output information about the query, whereas we use them directly as output scores because our task is referential. This also allows us to tie the keys to the characters of the Friends series as in the previous model, and thus have them represent entities (in the original model, the keys represented entity types, not instances).

4 Character Identification

The training and test data for the task span the first two seasons of *Friends*, divided into scenes and episodes, which were in turn divided into utterances (and tokens) annotated with speaker identity.[7] The set of all possible entities to refer to is given, as well as the set of mentions to resolve. Only the dialogues and speaker information are available (e.g., no video or descriptive text). Indeed,

[6]Two small changes with respect to the original model (motivated by empirical results in the hyperparameter search) are that we compute the gate using cosine similarity instead of dot product, and the obtained similarities are fed through a ReLU nonlinearity instead of sigmoid.

[7]The dataset also includes automatic linguistic annotations, e.g., PoS tags, which our models do not use.

models	#par	all (78) F$_1$	all (78) Acc	main (7) F$_1$	main (7) Acc
SemEv-1st	-	41.1	74.7	79.4	77.2
SemEv-2nd	-	13.5	68.6	83.4	82.1
biLSTM	3.4M	34.4	74.6	**85.0**	83.5
EntLib	3.3M	49.6*	**77.6***	84.9	**84.2**
EntNet	3.4M	**52.5***	77.5*	84.8	83.9

Table 1: Model parameters and results on the character identification task. First block: top systems at SemEval 2018. Results in the second block marked with * are statistically significantly better than biLSTM at $p < 0.001$ (approximate randomization tests, Noreen, 1989).

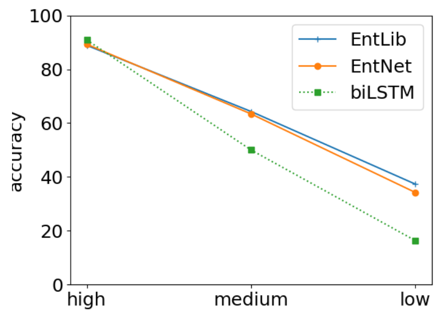

Figure 5: Accuracy on entities with high (>1000), medium (20–1000), and low (<20) frequency.

one of the most interesting aspects of the SemEval data is the fact that it is dialogue (even if scripted), which allows us to explore the role of speaker information, one of the aspects of the extralinguistic context of utterance that is crucial for reference. We additionally used the publicly available 300-dimensional word vectors that were pre-trained on a Google News corpus with the word2vec Skip-gram model (Mikolov et al., 2013a) to represent the input tokens. Entity (speaker/referent) embeddings were randomly initialized.

We train the models with backpropagation, using the standard negative log-likelihood loss function. For each of the three model architectures we performed a random search (> 1500 models) over the hyperparameters using cross-validation (see Appendix for details), and report the results of the best settings after retraining without cross-validation. The findings we report are representative of the model populations.

Results. We follow the evaluation defined in the SemEval task. Metrics are macro-average F$_1$-score (which computes the F$_1$-score for each entity separately and then averages these over all entities) and accuracy, in two conditions: *All entities*, with 78 classes (77 for entities that are mentioned in both training and test set of the SemEval Task, and one grouping all others), and *main entities*, with 7 classes (6 for the main characters and one for all the others). Macro-average F$_1$-score on all entities, the most stringent, was the criterion to define the leaderboard.

Table 1 gives our results in the two evaluations, comparing the models described in Section 3 to the best performing models in the SemEval 2018 Task 4 competition (Aina et al., 2018; Park et al., 2018). Recall that our goal in this paper is not to optimize performance, but to understand model behavior; however, results show that these models are worth analyzing, as that they outperform the state of the art. All models perform on a par on main entities, but entity-centric models outperform biLSTM by a substantial margin when all characters are to be predicted (the difference between EntLib and EntNet is not significant).

The architectures of EntLib and EntNet help with lower frequency characters, while not hurting performance on main characters. Indeed, Figure 5 shows that the accuracy of biLSTM rapidly deteriorates for less frequent entities, whereas EntLib and EntNet degrade more gracefully. Deep learning approaches are data-hungry, and entity mentions follow the Zipfian distribution typical of language, with very few high frequency and many lower-frequency items, such that this is a welcome result. Moreover, these improvements do not come at the cost of model complexity in terms of number of parameters, since all models have roughly the same number of parameters ($3.3 - 3.4$ million).[8]

Given these results and the motivations for the model architectures, it would be tempting to conclude that encouraging models to learn and use entity representations helps in this referential task. However, a closer look at the models' behavior reveals a much more nuanced picture.

Figure 6 suggests that: (1) models are quite good at using speaker information, as the best performance is for first person pronouns and determiners (*I*, *my*, etc.); (2) instead, models do not seem to be very good at handling other contextual information or entity-specific properties, as the worst

[8]See Appendix for a computation of the models' parameters.

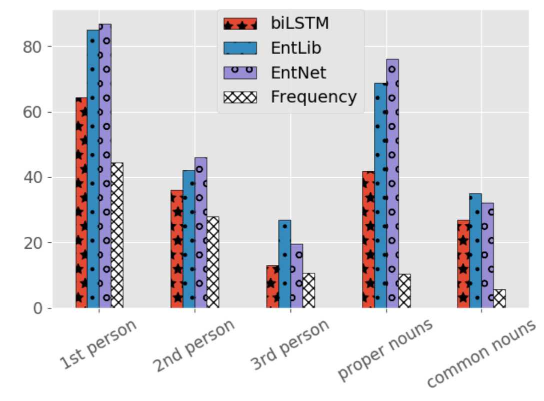

Figure 6: F_1-score (*all entities* condition) of the three models, per mention type, and token frequency of each mention type.

performance is for third person mentions and common nouns, which require both;[9] (3) EntLib and EntNet behave quite similarly, with performance boosts in (1) and smaller but consistent improvements in (2). Our analyses in the next two sections confirm this picture and relate it to the models' architectures.

5 Analysis: Architecture

We examine how the entity-centric architectures improve over the biLSTM baseline on the reference task, then move to entity representations (Section 6).

Shared speaker/referent representation. We found that an important advantage of the entity-centric models, in particular for handling low-frequency entities, lies in the integrated representations they enable of entities both in their role of speakers and in their role of referents. This explains the boost in first person pronoun and proper noun mentions, as follows.

Recall that the integrated representation is achieved by parameter sharing, using the same weight matrix $\mathbf{W}_e$ as speaker embedding and as entity library/keys. This enables entity-centric models to learn the linguistic rule "a first person pronoun (*I, me*, etc.) refers to the speaker" regardless of whether they have a meaningful representation of this particular entity: It is enough that speaker representations are distinct, and they are because they have been randomly initialized. In contrast, the simple biLSTM baseline needs to independently learn the mapping between speaker embedding and output entities, and so it can only learn to resolve even first-person pronouns for entities for which it has enough data.

[9] 1st person: *I*, *me*, *my*, *myself*, *mine*; 2nd person: *you*, *your*, *yourself*, *yours*; 3rd person: *she*, *her*, *herself*, *hers*, *he*, *him*, *himself*, *his*, *it*, *itself*, *its*.

model type	main	all
biLSTM	0.39	0.02
EntLib	0.82	0.13
EntNet	0.92	0.16
#pairs	21	22155

Table 2: RSA correlation between speaker/referent embeddings W_e and token embeddings W_t of the entities' names, for main entities vs. all entities (right)

For proper nouns (character names), entity-centric models learn to align the token embeddings with the entity representations (identical to the speaker embeddings). We show this by using Representation Similarity Analysis (RSA) (Kriegeskorte et al., 2008), which measures how topologically similar two different spaces are as the Spearman correlation between the pair-wise similarities of points in each space (this is necessary because entities and tokens are in different spaces). For instance, if the two spaces are topologically similar, the relationship of entities 183 and 335 in the entity library will be analogous to the relationship between the names *Joey* and *Ross* in the token space. Table 2 shows the topological similarities between the two spaces, for the different model types.[10] This reveals that in entity-centric models the space of speaker/referent embeddings is topologically very similar to the space of token embeddings restricted to the entities' names, and more so than in the biLSTM baseline. We hypothesize that entity-centric models can do the alignment better because referent (and hence speaker) embeddings are closer to the error signal, and thus backpropagation is more effective (this again helps with lower-frequency entities).

Further analysis revealed that in entity-centric models the beneficial effect of weight sharing between the speaker embedding and the entity representations (both $\mathbf{W}_e$) is actually restricted to first-person pronouns. For other expressions, having

[10] As an entity's name we here take the proper noun that is most frequently used to refer to the entity in the training data. Note that for the *all entities* condition the absolute values are lower, but the space is much larger (over 22K pairs). Also note that this is an instance of slow learning; models are not encoding the fact that a proper noun like *Rachel* can refer to different people.

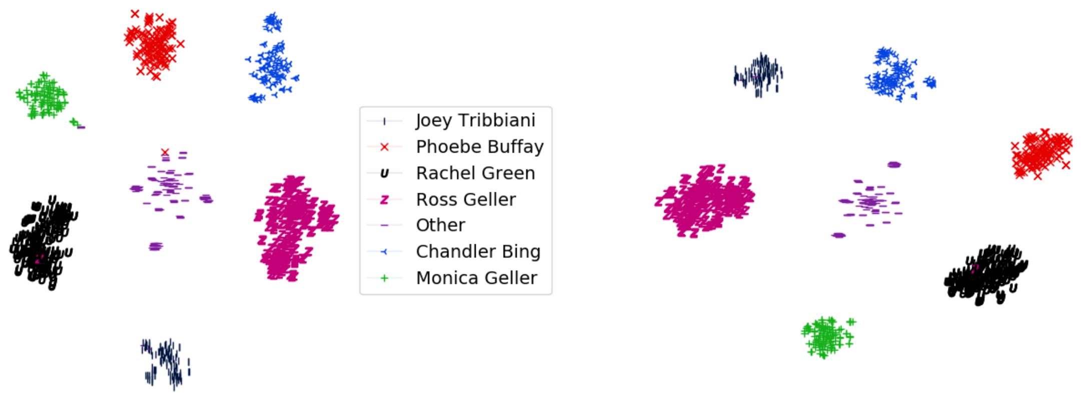

Figure 7: ENTLIB, 2D TSNE projections of the activations for first-person mentions in the test set, colored by the entity referred to. The mentions cluster into entities already in the hidden layer h_i (left graph; query layer q_i shown in the right graph). Best viewed in color.

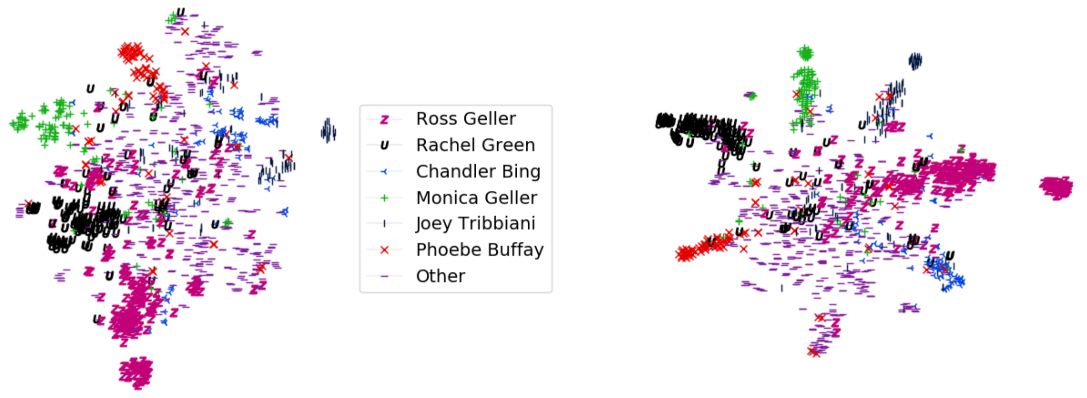

Figure 8: ENTLIB, 2D TSNE projections of the activations for mentions in the test set (excluding first person mentions), colored by the entity referred to. While there is already some structure in the hidden layer h_i (left graph), the mentions cluster into entities much more clearly in the query q_i (right graph). Best viewed in color.

two distinct matrices yielded almost the same performance as having one (but still higher than the BILSTM, thanks to the other architectural advantage that we discuss below).

In the case of first-person pronouns, the speaker embedding given as input corresponds to the target entity. This information is already accessible in the hidden state of the LSTM. Therefore, mentions cluster into entities already at the hidden layer h_i, with no real difference with the query layer q_i (see Figure 7).

Advantage of query layer. The entity querying mechanism described above entails having an extra transformation after the hidden layer, with the query layer $\mathbf{q}$. Part of the improved performance of entity-centric models, compared to the BILSTM baseline, is due not to their bias towards 'entity representations' per se, but due to the presence of this extra layer. Recall that the BILSTM baseline maps the LSTM's hidden state $\mathbf{h}_i$ to output scores $\mathbf{o}_i$ with a single transformation. Gulordava et al. (2018a) observe in the context of Language Modeling that this creates a tension between two conflicting requirements for the LSTM: keeping track of contextual information across time steps, and encoding information useful for prediction in the current timestep. The intermediate query layer $\mathbf{q}$ in entity-centric models alleviates this tension. This explains the improvements in context-dependent mentions like common nouns or second and third pronouns.

We show this effect in two ways. First, we compare the average mean similarity s of mention pairs $T_e = \{(t_k, t_{k'}) | \; t_k \rightarrow e \wedge k \neq k'\}$ referring to the same entity e in the hidden layer (Eq. 13) and the

BILSTM	ENTLIB		ENTNET	
$\mathbf{h}_i$	$\mathbf{h}_i$	$\mathbf{q}_i$	$\mathbf{h}_i$	$\mathbf{q}_i$
0.34	0.24	0.48	0.27	0.60

Table 3: Average cosine similarity of mentions with the same referent.

query layer.[11]

$$s = \frac{1}{|E|}\sum_{e\in E}\frac{1}{|T_e|}\sum_{(t_k,t_{k'})\in T_e}\cos(h_{t_k},h_{t_{k'}}) \quad (13)$$

Table 3 shows that, in entity-centric models, this similarity is lower in the hidden layer $\mathbf{h}_i$ than in the case of the BILSTM baseline, but in the query layer $\mathbf{q}_i$ it is instead much higher. The hidden layer thus is representing other information than referent-specific knowledge, and the query layer can be seen as extracting referent-specific information from the hidden layer. Figure 8 visually illustrates the division of labor between the hidden and query layers. Second, we compared the models to variants where the cosine-similarity comparison is replaced by an ordinary dot-product transformation, which converts the querying mechanism into a simple further layer. These variants performed almost as well on the reference task, albeit with a slight but consistent edge for the models using cosine similarity.

No dynamic updates in ENTNET. A surprising negative finding is that ENTNET is not using its dynamic potential on the referential task. We confirmed this in two ways. First, we tracked the values $\mathbf{V}_i$ of the entity representations and found that the pointwise difference in $\mathbf{V}_i$ at any two adjacent time steps i tended to zero. Second, we simply switched off the update mechanism during testing and did not observe any score decrease on the reference task. ENTNET is thus only using the part of the entity memory that it shares with ENTLIB, i.e., the keys $\mathbf{W_e}$, which explains their similar performance.

This finding is markedly different from Henaff et al. (2017), where for instance the BaBI tasks could be solved only by dynamically updating the entity representations. This may reflect our different language modules: since our LSTM module already has a form of dynamic memory, unlike the simpler sentence processing module in Henaff et al. (2017), it may be that the LSTM takes this burden off of the entity module. An alternative is that it is due to differences in the datasets.

[11]For the query layer, Eq. 13 is equivalent, with $\cos(q_{t_k}, q_{t_{k'}})$.

This person is {a/an/the} <PROPERTY> [and {a/an/the} <PROPERTY>]{0,2}.

This person is the brother of Monica Geller.
This person is a paleontologist and a man.

Figure 9: Patterns and examples (in italics) of the dataset for information extraction as entity linking.

We leave an empirical comparison of these potential explanations for future work, and focus in Section 6 on the static entity representations $\mathbf{W}_e$ that ENTNET essentially shares with ENTLIB.

6 Analysis: Entity Representations

The foregoing demonstrates that entity-centric architectures help in a reference task, but not that the induced representations in fact contain meaningful entity information. In this section we deploy these representations on a new dataset, showing that they do not—not even for basic information about entities such as gender.

Method. We evaluate entity representations with an information extraction task including attributes and relations, using information from an independent, unstructured knowledge base—the Friends Central Wikia.[12] To be able to use the models as is, we set up the task in terms of entity linking, asking models to solve the reference of natural language descriptions that uniquely identify an entity. For instance, given *This person is the brother of Monica Geller.*, the task is to determine that *person* refers to *Ross Geller*, based on the information in the sentence.[13] The information in the descriptions was in turn extracted from the Wikia. We do not retrain the models for this task in any way—we simply deploy them.

We linked the entities from the Friends dataset used above to the Wikia through a semi-automatic procedure that yielded 93 entities, and parsed the Wikia to extract their attributes (*gender* and *job*) and relations (e.g., *sister*, *mother-in-law*; see Appendix for details). We automatically generate the natural language descriptions with a simple pattern (Figure 9) from combinations of properties that uniquely identify a given entity within the set of Friends characters.[14] We

[12]http://friends.wikia.com.

[13]The referring expression is the whole DP, *This person*, but we follow the method in Aina et al. 2018 of asking for reference resolution at the head noun.

[14]Models require inputting a speaker; we use speaker UN-

model	description	gender	job
RANDOM	1.5	50	20
BILSTM	0.4	-	-
ENTLIB	2.2	55	27
ENTNET	1.3	61	24

Table 4: Results on the attribute and relation prediction task: percentage accuracy for natural language descriptions, mean reciprocal rank of characters for single attributes (lower is worse).

consider unique descriptions comprising at most 3 properties. Each property is expressed by a noun phrase, whereas the article is adapted (definite or indefinite) depending on whether that property applies to one or several entities in our data. This yields 231 unique natural language descriptions of 66 characters, created on the basis of overall 61 relation types and 56 attribute values.

Results. The results of this experiment are negative: The first column of Table 4 shows that models get accuracies near 0.

A possibility is that models do encode information in the entity representations, but it doesn't get used in this task because of how the utterance is encoded in the hidden layer, or that results are due to some quirk in the specific setup of the task. However, we replicated the results in a setup that does not encode whole utterances but works with single attributes and relations. While the methodological details are in the Appendix, the 'gender' and 'job' columns of Table 4 show that results are a bit better in this case but models still perform quite poorly: Even in the case of an attribute like gender, which is crucial for the resolution of third person pronouns (*he/she*), the models' results are quite close to that of a random baseline.

Thus, we take it to be a robust result that entity-centric models trained on the SemEval data do not learn or use entity information—at least as recoverable from language cues. This, together with the remainder of the results in the paper, suggests that models rely crucially on speaker information, but hardly on information from the linguistic context.[15] Future work should explore alternatives such as pre-training with a language modeling task, which could improve the use of context.

KNOWN.

[15]Note that 44% of the mentions in the dataset are first person, for which linguistic context is irrelevant and the models only need to recover the relevant speaker embedding to succeed. However, downsampling first person mentions did not improve results on the other mention types.

7 Conclusions

Recall that the motivation for entity-centric models is the hypothesis that incorporating entity representations into the model will help it better model the language we use to talk about them. We still think that this hypothesis is plausible. However, the architectures tested do not yet provide convincing support for it, at least for the data analyzed in this paper.

On the positive side, we have shown that framing models from an entity-centric perspective makes it very natural to adopt architectural decisions that are good inductive biases. In particular, by exploiting the fact that both speakers and referents are entities, these models can do more with the same model size, improving results on less frequent entities and emulating rule-based behavior such as "a first person expression refers to the speaker". On the negative side, we have also shown that they do not yield operational entity representations, and that they are not making good use of contextual information for the referential task.

More generally, our paper underscores the need for model analysis to test whether the motivations for particular architectures are borne out in how the model actually behaves when it is deployed.

Acknowledgments

We gratefully acknowledge Kristina Gulordava and Marco Baroni for the feedback, advice and support. We are also grateful to the anonymous reviewers for their valuable comments. 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 715154), and from the Spanish Ramón y Cajal programme (grant RYC-2015-18907). We are grateful to the NVIDIA Corporation for the donation of GPUs used for this research. We are also very grateful to the Pytorch developers. This paper reflects the authors' view only, and the EU is not responsible for any use that may be made of the information it contains.

References

Laura Aina, Carina Silberer, Ionut-Teodor Sorodoc, Matthijs Westera, and Gemma Boleda. 2018. AMORE-UPF at SemEval-2018 Task 4: BiLSTM with Entity Library. In *Proc. of SemEval*.

Trapit Bansal, Arvind Neelakantan, and Andrew McCallum. 2017. RelNet: End-to-End Modeling of Entities & Relations. *CoRR*, abs/1706.07179.

Razvan Bunescu and Marius Paşca. 2006. Using Encyclopedic Knowledge for Named Entity Disambiguation. In *Proc. of EACL*.

Henry Y. Chen, Ethan Zhou, and Jinho D. Choi. 2017. Robust Coreference Resolution and Entity Linking on Dialogues: Character Identification on TV Show Transcripts. In *Proc. of CoNLL 2017*.

Hui Chen, Baogang Wei, Yonghuai Liu, Yiming Li, Jifang Yu, and Wenhao Zhu. 2018. Bilinear Joint Learning of Word and Entity Embeddings for Entity Linking. *Neurocomputing*, 294:12–18.

Jinho D Choi and Henry Y Chen. 2018. SemEval 2018 Task 4: Character Identification on Multiparty Dialogues. In *Proc. of SemEval*.

Elizabeth Clark, Yangfeng Ji, and Noah A Smith. 2018. Neural Text Generation in Stories Using Entity Representations as Context. In *Proc. of NAACL*.

Kevin Clark and Christopher D. Manning. 2016. Improving Coreference Resolution by Learning Entity-Level Distributed Representations. In *Proc. of ACL*.

Alexis Conneau, Germán Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What You Can Cram into a Single $&!#* Vector: Probing Sentence Embeddings for Linguistic Properties. In *Proc. of ACL*.

Nick Craswell. 2009. Mean Reciprocal Rank. In *Encyclopedia of Database Systems*, pages 1703–1703. Springer.

Matthew Francis-Landau, Greg Durrett, and Dan Klein. 2016. Capturing Semantic Similarity for Entity Linking with Convolutional Neural Networks. In *Proc. of NAACL:HLT*.

Mario Giulianelli, Jack Harding, Florian Mohnert, Dieuwke Hupkes, and Willem Zuidema. 2018. Under the Hood: Using Diagnostic Classifiers to Investigate and Improve how Language Models Track Agreement Information. In *Proc. of EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*.

Alex Graves, Santiago Fernández, and Jürgen Schmidhuber. 2005. Bidirectional LSTM Networks for Improved Phoneme Classification and Recognition. In *Proc. of ICANN*.

Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural Turing Machines. *CoRR*, abs/1410.5401.

Kristina Gulordava, Laura Aina, and Gemma Boleda. 2018a. How to Represent a Word and Predict it, too: Improving Tied Architectures for Language Modelling. In *Proc. of EMNLP*.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018b. Colorless Green Recurrent Networks Dream Hierarchically. In *Proc. of NAACL*.

Aria Haghighi and Dan Klein. 2010. Coreference Resolution in a Modular, Entity-Centered Model. In *Proc. of NAACL*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In *Proc. of ICCV*.

Mikael Henaff, Jason Weston, Arthur Szlam, Antoine Bordes, and Yann LeCun. 2017. Tracking the World State with Recurrent Entity Networks. In *Proc. of ICLR*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Armand Joulin and Tomas Mikolov. 2015. Inferring Algorithmic Patterns with Stack-Augmented Recurrent Nets. In *Proc. of NIPS*.

Akos Kádár, Grzegorz Chrupała, and Afra Alishahi. 2017. Representation of Linguistic Form and Function in Recurrent Neural Networks. *Computational Linguistics*, 43(4):761–780.

Hans Kamp and Uwe Reyle. 2013. *From Discourse to Logic: Introduction to Model-theoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*, volume 42. Springer Science+Business Media.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR*, abs/1412.6980.

Nikolaus Kriegeskorte, Marieke Mur, and Peter A Bandettini. 2008. Representational Similarity Analysis – Connecting the Branches of Systems Neuroscience. *Frontiers in Systems Neuroscience*, 2:4.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the Ability of LSTMs to Learn Syntax-Sensitive Dependencies. *TACL*, 4(1):521–535.

Rada Mihalcea and Andras Csomai. 2007. Wikify!: Linking Documents to Encyclopedic Knowledge. In *Proc. of CIKM*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Distributed Representations of Words and Phrases and Their Compositionality. In *Proc. of NIPS*.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013b. Linguistic Regularities in Continuous Space Word Representations. In *Proc. of NAACL*.

Aida Nematzadeh, Kaylee Burns, Erin Grant, Alison Gopnik, and Tom Griffiths. 2018. Evaluating Theory of Mind in Question Answering. In *Proc. of EMNLP*.

E.W. Noreen. 1989. *Computer-Intensive Methods for Testing Hypotheses: An Introduction*. Wiley.

Cheon-Eum Park, Heejun Song, and Changki Lee. 2018. KNU CI System at SemEval-2018 Task4: Character Identification by Solving Sequence-Labeling Problem. In *Proc. of SemEval*.

Sameer Pradhan, Lance Ramshaw, Mitchell Marcus, Martha Palmer, Ralph Weischedel, and Nianwen Xue. 2011. CoNLL-2011 Shared Task: Modeling Unrestricted Coreference in OntoNotes. In *Proc. of CoNLL*.

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-End Memory Networks. In *Proc. of NIPS*.

Sam Wiseman, Alexander M Rush, and Stuart M Shieber. 2016. Learning Global Features for Coreference Resolution. In *Proc. of NAACL:HLT*.

Zichao Yang, Phil Blunsom, Chris Dyer, and Wang Ling. 2017. Reference-Aware Language Models. In *Proc. of EMNLP*.

A Appendices

A.1 Hyperparameter search

Besides the LSTM parameters, we optimize the token embeddings $\mathbf{W}_t$, the entity/speaker embeddings $\mathbf{W}_e$, as well as $\mathbf{W}_o$, $\mathbf{W}_q$, and their corresponding biases, where applicable (see Section 3). We used five-fold cross-validation with early stopping based on the validation score. We found that most hyperparameters could be safely fixed the same way for all three types. Specifically, our final models were all trained in batch mode using the Adam optimizer (Kingma and Ba, 2014), with each batch covering 25 scenes given to the model in chunks of 750 tokens. The token embeddings ($\mathbf{W}_t$) are initialized with the 300-dimensional word2vec vectors, $\mathbf{h}_i$ is set to 500 units, and entity (or speaker) embeddings ($\mathbf{W}_e$) to $k = 150$ units.With this hyperparameter setting, ENTLIB has fewer parameters than BILSTM: the linear map $\mathbf{W}_o$ of the latter (500×401) is replaced by the query extractor $\mathbf{W}_q$ (500×150) followed by (non-parameterized) similarity computations. This holds even if we take into account that the entity embedding $\mathbf{W}_e$ used in both models contains 274 entities that are never speakers and that are, hence, used by ENTLIB but not by BILSTM.

Our search also considered different types of activation functions in different places, with the architecture presented above, i.e., tanh before the LSTM and ReLU in the gate, robustly yielding the best results. Other settings tested—randomly initialized token embeddings, self-attention on the input layer, and a uni-directional LSTM—did not improve performance.

We then performed another random search (> 200 models) over the remaining hyperparameters: learning rate (sampled from the logarithmic interval 0.001–0.05), dropout before and after LSTM (sampled from 0.0–0.3 and 0.0–0.1, respectively), weight decay (sampled from 10^{-6}–10^{-2}) and penalization, i.e., whether to decrease the relative impact of frequent entities by dividing the loss for an entity by the square root of its frequency. This paper reports the best model of each type, i.e., BILSTM, ENTLIB, and ENTNET, after training on all the training data without cross-validation for 20, 80 and 80 epochs respectively (numbers selected based on tendencies in training histories). These models had the following parameters:

	BILSTM	ENTLIB	ENTNET
learning rate:	0.0080	0.0011	0.0014
dropout pre	0.2	0.2	0.0
dropout post:	0.0	0.02	0.08
weight decay:	1.8e-6	4.3e-6	1.0e-5
penalization:	no	yes	yes

B Attribute and relation extraction

B.1 Details of the dataset

We performed a two-step procedure to extract all the available data for the SemEval characters. First, using simple word overlap, we automatically mapped the 401 SemEval names to the characters in the database. In a second, manual step, we corrected these mappings and added links that were not found automatically due to name alternatives, ambiguities or misspellings (e.g., SemEval *Dana* was mapped to *Dana Keystone*, and *Janitor* to *The Zoo Employee*). In total, we found 93 SemEval entities in Friends Central, and we extracted their attributes (gender and job) and their mutual relationships (relatives).

Model	Gender (93;2) (wo)man	(s)he	Occupation (24;17)	Relatives (56;24)
RANDOM	.50	.50	.20	.16
ENTLIB	.55	.58	.27	.22
ENTNET	**.61**	.56	.24	**.26**

Table 5: Results on the attribute prediction task (mean reciprocal rank; from 0 (worst) to 1 (best)). The number of considered test items and candidate values, respectively, are given in the parentheses. For gender, we used *(wo)man* and *(s)he* as word cues for the values (fe)male.

B.2 Alternative setup

We use the same models, i.e. ENTLIB and ENTNET trained on Experiment 1, and (without further training) extract representations for the entities from them. The former are directly obtained from the entity embedding $\mathbf{W}_e$ of each model.

In the **attribute prediction** task, we are given an attribute (e.g., *gender*), and all its possible values $\mathcal{V}$ (e.g., $\mathcal{V} = \{$*woman, man* $\}$). We formulate the task as, given a character (e.g., *Rachel*), producing a ranking of the possible values in descending order of their similarity to the character, where similarity is computed by measuring the cosine of the angle between their respective vector representations in the entity space. We obtain representations of attributes values, in the same space as the entities, by inputting each attribute value as a separate utterance to the models, and extracting the corresponding entity query ($\mathbf{q}_i$). Since the models also expect a speaker for each utterance, we set it to either *all entities*, *main entities*, a *random* entity, or *no* entity (i.e., speaker embedding with zero in all units), and report the best results.

We evaluate the rankings produced for both tasks in terms of mean reciprocal rank (Craswell, 2009), scoring from 0 to 1 (from worst to best) the position of the target labels in the ranking. The two first columns Table 5 presents the results. Our models generally perform poorly on the tasks, though outperforming a random baseline. Even in the case of an attribute like gender, which is crucial for the resolution of third person pronouns, the models' results are still very close to that of the random baseline.

Instead, the task of **relation prediction** is to, given a pair of characters (e.g., *Ross* and *Monica*), predict the relation R which links them (e.g., *sister, brother-in-law, nephew*; we found 24 relations that applied to at least two pairs). We approach this following the vector offset method introduced by Mikolov et al. (2013b) for semantic relations between words. This leverages on regularities in the embedding space, taking the embeddings of pairs that are connected by the same relation to have analogous spatial relations. For two pairs of characters (a, b) and (c, d) which bear the same relation R, we assume $\mathbf{a} - \mathbf{b} \approx \mathbf{c} - \mathbf{d}$ to hold for their vector representations. For a target pair (a, b) and a relation R, we then compute the following measure:

$$\mathrm{s}_{rel}((a, b), R) = \frac{\sum_{(x,y)\in R} \cos(\mathbf{a} - \mathbf{b}, \mathbf{x} - \mathbf{y})}{|R|} \tag{14}$$

Equation (14) computes the average relational similarity between the target character pair and the exemplars of that relation (excluding the target itself), where the relational similarity is estimated as the cosine between the vector differences of the two pairs of entity representations respectively. Due to this setup, we restrict to predicting relation types that apply to at least two pairs of entities. For each target pair (a, b), we produce a rank of candidate relations in descending order of their scores s_{rel}. Table 5 contains the results, again above baseline but clearly very poor.