Emergent Predication Structure in Hidden State Vectors of Neural Readers

Hai Wang* **Takeshi Onishi*** **Kevin Gimpel** **David McAllester**
Toyota Technological Institute at Chicago
6045 S. Kenwood Ave., Chicago, Illinois 60637, USA
{haiwang,tonishi,kgimpel,mcallester}@ttic.edu

Abstract

A significant number of neural architectures for reading comprehension have recently been developed and evaluated on large cloze-style datasets. We present experiments supporting the emergence of "predication structure" in the hidden state vectors of these readers. More specifically, we provide evidence that the hidden state vectors represent atomic formulas $\Phi[c]$ where Φ is a semantic property (predicate) and c is a constant symbol entity identifier.

1 Introduction

Reading comprehension is a type of question answering task where the answer is to be found in a passage about particular entities and events. In particular, the entities and events should not be mentioned in structured databases of general knowledge. Reading comprehension problems are intended to measure a system's ability to extract semantic information about entities and relations directly from unstructured text.

Several large scale reading comprehension datasets have been introduced recently, including the CNN & Daily Mail datasets (Hermann et al., 2015), the Children's Book Test (CBT) (Hill et al., 2016), and the Who-did-What dataset (Onishi et al., 2016). The large sizes of these datasets enable the application of deep learning. These are all cloze-style datasets where a question is constructed by deleting a word or phrase from an article summary (in CNN/Daily Mail), from a sentence in a children's story (in CBT), or by deleting a person from the first sentence of a different news article on the same entities and events (in Who-did-What).

In this paper we present empirical evidence for the emergence of predication structure in a certain class of neural readers. To understand predication structure, it is helpful to review the anonymization performed in the CNN/Daily Mail dataset. In this dataset named entities are replaced by anonymous entity identifiers such as "entity37". The passage might contain "entity52 gave entity24 a rousing applause" and the question might be "X received a rounding applause from entity52". The task is to fill in X from a given multiple choice list of candidate entity identifiers. A fixed relatively small set of the same entity identifiers are used over all the problems and the same problem is presented many times with different entity identifiers shuffled. This prevents a given entity identifier from having any semantically meaningful vector embedding. The embeddings of the entity identifiers are presumably just pointers to semantics-free tokens. We will write entity identifiers as logical constant symbols such as c rather than strings such as "entity37".

"Aggregation" readers, including Memory Networks (Weston et al., 2015; Sukhbaatar et al., 2015), the Attentive Reader (Hermann et al., 2015), and the Stanford Reader (Chen et al., 2016), use bidirectional LSTMs or GRUs to construct a contextual embedding h_t of each position t in the passage and also an embedding h_q of the question q. They then select an answer c using a criterion similar to

$$\operatorname*{argmax}_c \sum_t < h_t, h_q > \; < h_t, e(c) > \qquad (1)$$

where $e(c)$ is the vector embedding of the constant symbol (entity identifier) c. In practice the inner-product $< h_t, h_q >$ is normalized over t using a softmax to yield attention weights α_t over t and

* Authors contributed equally.

Proceedings of the 2nd Workshop on Representation Learning for NLP, pages 26–36,
Vancouver, Canada, August 3, 2017.

(1) becomes

$$\underset{c}{\text{argmax}} \; < e(c), \sum_t \alpha_t h_t > . \qquad (2)$$

Here $\sum_t \alpha_t h_t$ can be viewed as a vector representation of the passage.

We argue that for aggregation readers, roughly defined by (2), the hidden state h_t of the passage at position (or word) t can be viewed as a vector concatenation $h_t = [s(\Phi_t), s(c_t)]$ where Φ_t is a property (or statement or predicate) being stated of a particular constant symbol c_t. Here $s(\Phi_t)$ and $s(c_t)$ are unknown emergent embeddings of Φ_t and c_t respectively. A logician might write this as $h_t = \Phi_t[c_t]$. Furthermore, the question can be interpreted as having the form $\Psi[x]$ where the problem is to find a constant symbol c such that the passage implies $\Psi[c]$. Assuming $h_t = [s(\Phi_t), s(c_t)]$, $h_q = [s(\Psi), 0]$, and $e(c) = [0, s(c)]$, we can rewrite (1) as

$$\underset{c}{\text{argmax}} \sum_t < s(\Phi_t), s(\Psi) > \; < s(c_t), s(c) > . \qquad (3)$$

The first inner product in (3) is interpreted as measuring the extent to which $\Phi_t[x]$ implies $\Psi[x]$ for any x. The second inner product is interpreted as restricting t to positions talking about the constant symbol c.

Note that the posited decomposition of h_t is not explicit in (2) but instead must emerge during training. We present empirical evidence that this structure does emerge. The empirical evidence is somewhat tricky as the direct sum structure that divides h_t into its two parts need not be axis aligned and therefore need not literally correspond to vector concatenation.

We also consider a second class of neural readers that we call "explicit reference" readers. Explicit reference readers avoid (2) and instead use

$$\underset{c}{\text{argmax}} \sum_{t \in R(c)} \alpha_t \qquad (4)$$

where $R(c)$ is the subset of the positions where the constant symbol (entity identifier) c occurs. Note that if we identify α_t with $< s(\Phi_t), s(\Psi) >$ and assume that $< s(c), s(c_t) >$ is either 0 or 1 depending on whether $c = c_t$, then (3) and (4) agree. In explicit reference readers the hidden state h_t need not carry a pointer to c_t as the restriction on t is independent of learned representations. Explicit reference readers include the Attention Sum Reader (Kadlec et al., 2016), the Gated Attention Reader (Dhingra et al., 2017), and the Attention-over-Attention Reader (Cui et al., 2017).

So far we have only considered anonymized datasets that require the handling of semantics-free constant symbols. However, even for non-anonymized datasets such as Who-did-What, it is helpful to add features which indicate which positions in the passage are referring to which candidate answers. This indicates, not surprisingly, that reference is important in question answering. The fact that explicit reference features are needed in aggregation readers on non-anonymized data indicates that reference is not being solved by the aggregation readers. However, as reference seems to be important for cloze-style question answering, these problems may ultimately provide training data from which reference resolution can be learned.

Sections 2 and 3 review various existing datasets and models respectively. In the CNN dataset the vector embeddings of entity identifiers such as "entity32" are clearly interpretable as vector representations of semantics-free constant symbols. However, to the best of our knowledge the emergent decomposition of the hidden state vectors into a concatenation of a property vector and an entity vector has not been previously described or empirically investigated in the literature. Section 4 presents the logical structure interpretation of aggregation readers in more detail and the empirical evidence supporting it. Section 5 proposes new models that enforce the direct sum structure of the hidden state vectors. It is shown that these new models perform well on the Who-did-What dataset provided that reference annotations are added as input features. Section 5 also describes additional linguistic features that can be added to the input embeddings and show that these improve the performance of existing models resulting in the best single-model performance to date on the Who-did-What dataset.

2 A Brief Survey of Datasets

Before presenting various models for machine comprehension we give a general formulation of the machine comprehension task. We take an instance of the task to be a four tuple $(q, p, a, \mathcal{A})$, where q is a question given as a sequence of words containing a special token for a "blank" to be filled in, p is a document consisting of a sequence of

words, $\mathcal{A}$ is a set of possible answers and $a \in \mathcal{A}$ is the ground truth answer. All words are drawn from a vocabulary $\mathcal{V}$. We assume that all possible answers are words from the vocabulary, that is $\mathcal{A} \subseteq \mathcal{V}$, and that the ground truth answer appears in the document, that is $a \in p$. The problem can be described as that of selecting the answer $a \in \mathcal{A}$ that answers question q based on information from p. We now briefly summarize important features of the related datasets in reading comprehension.

CNN & Daily Mail: Hermann et al. (2015) constructed these datasets from a large number of news articles from the CNN and Daily Mail news websites. The main article is used as the context, while the cloze style question is formed from one short article summary sentence appearing in conjunction with the published article. To avoid the model using external world knowledge when answering the question, the named entities in the entire dataset were replaced by anonymous entity IDs which were then further shuffled for each example. This forces models to rely on the context document to answer each question. In this anonymized corpus the entity identifiers are taken to be a part of the vocabulary and the answer set $\mathcal{A}$ consists of the entity identifiers occurring in the passage.

Who-did-What (WDW): The Who-did-What dataset (Onishi et al., 2016) contains 127,000 multiple choice cloze questions constructed from the LDC English Gigaword newswire corpus (David and Cieri, 2003). In contrast with CNN and Daily Mail, WDW avoids using article summaries for question formation. Instead, each problem is formed from two independent articles: one is given as the passage to be read and a different article on the same entities and events is used to form the question. Further, WDW avoids anonymization — each choice is a person named entity. In this dataset the answer set $\mathcal{A}$ consists of the person named entities occurring in the passage. Finally, the problems have been filtered to remove a fraction that are easily solved by simple baselines. It has two training sets. The larger training set ("relaxed") is created using less baseline filtering, while the smaller training set ("strict") uses the same filtering as the validation and test sets.

Other Related Datasets. It is also worth mentioning several related datasets. The MCTest dataset (Richardson et al., 2013) consists of children's stories and questions written by crowdsourced workers. The dataset only contains 660 documents and is too small to train deep models. The bAbI dataset (Weston et al., 2016) is constructed automatically using synthetic text generation and can be perfectly answered by hand-written algorithms (Lee et al., 2016). The SQuAD dataset (Rajpurkar et al., 2016) consists of passage-question pairs where the passage is a Wikipedia article and the questions are written via crowdsourcing. The dataset contains over 100,000 problems, but the answer is often a word sequence which is difficult to handle with the reader models considered here. The Children's Book Test (CBT) (Hill et al., 2016) takes any sequence of 21 consecutive sentences from a children's book: the first 20 sentences are used as the passage, and the goal is to infer a missing word in the 21st sentence. The task complexity varies with the type of the omitted word (verb, preposition, named entity, or common noun). The LAMBADA dataset (Paperno et al., 2016) is a word prediction dataset which requires a broad discourse context, though the correct answer might not actually be contained in the context. Nevertheless, when the correct answer is in the context, neural readers can be applied effectively (Chu et al., 2017).

3 Aggregation Readers and Explicit Reference Readers

As outlined in the introduction, here we classify readers into aggregation readers and explicit reference readers. Aggregation readers appeared first in the literature and include Memory Networks (Weston et al., 2015; Sukhbaatar et al., 2015), the Attentive Reader (Hermann et al., 2015), and the Stanford Reader (Chen et al., 2016). In this section we define aggregation readers more specifically by equations (7) and (9) below. Explicit reference readers include the Attention-Sum Reader (Kadlec et al., 2016), the Gated-Attention Reader (Dhingra et al., 2017), and the Attention-over-Attention Reader (Cui et al., 2017). In this section we define explicit reference readers more specifically by equation (13) below. We first present the Stanford Reader as a paradigmatic aggregation reader and the Attention-Sum Reader as a paradigmatic explicit reference reader.

3.1 Aggregation Readers

Stanford Reader. The Stanford Reader (Chen et al., 2016) computes a bidirectional LSTM representation of both the passage and the question.

$$h = \text{biLSTM}(e(p)) \quad (5)$$

$$h_q = [\text{fLSTM}(e(q))_{|q|}, \text{bLSTM}(e(q))_1] \quad (6)$$

In equations (5) and (6) we have that $e(p)$ is the sequence of word embeddings $e(w_i)$ for $w_i \in p$ and similarly for $e(q)$. The expression biLSTM(s) denotes the sequence of hidden state vectors resulting from running a bidirectional LSTM on the vector sequence s. We write biLSTM$(s)_i$ for the ith vector in this sequence. Similarly fLSTM(s) and bLSTM(s) denote the sequence of vectors resulting from running a forward LSTM and a backward LSTM respectively and $[\cdot,\cdot]$ denotes vector concatenation. The Stanford Reader, and various other readers, then compute a bilinear attention over the passage which is used to construct a single weighted vector representation of the passage.

$$\alpha_t = \underset{t}{\text{softmax}}\ h_t^\top W_\alpha\, h_q \quad o = \sum_t \alpha_t h_t \quad (7)$$

Finally, they compute a probability distribution P over the answers:

$$P(\cdot|d,q,\mathcal{A}) = \underset{a\in\mathcal{A}}{\text{softmax}}\ e_o(a)^\top o \quad (8)$$

$$\hat{a} = \underset{a\in\mathcal{A}}{\text{argmax}}\ e_o(a)^\top o \quad (9)$$

Here $e_o(a)$ is the "output embedding" of the answer a. On the CNN dataset the Stanford Reader trains an output embedding for each of the roughly 550 entity identifiers used in the dataset. For datasets in which the answer might be any word in $\mathcal{V}$, output embeddings must be trained for the entire vocabulary.

The reader is trained with log-loss $-\log P(a|p,q,\mathcal{A})$ where a is the correct answer. At test time the reader is scored on the percentage of problems where $\hat{a} = a$.

Memory Networks. Memory Networks (Weston et al., 2015; Sukhbaatar et al., 2015) use (7) and (9) but have more elaborate methods of constructing "memory vectors" h_t not involving LSTMs. Memory networks use (7) and (9) but replace (8) with

$$P(\cdot|p,q,\mathcal{A}) = P(\cdot|p,q) = \underset{w\in\mathcal{V}}{\text{softmax}}\, e_o(w)^\top o. \quad (10)$$

It should be noted that (10) trains output vectors over the whole vocabulary rather than just those items occurring in the choice set $\mathcal{A}$. This is empirically significant in non-anonymized datasets such as CBT and Who-did-What where choices at test time may never have occurred as choices in the training data.

Attentive Reader. The Stanford Reader was derived from the Attentive Reader (Hermann et al., 2015). The Attentive Reader uses $\alpha_t = \text{softmax}_t \text{MLP}([h_t, h_q])$ instead of (7). Here MLP(x) is the output of a multi layer perceptron given input x. Also, the answer distribution in the Attentive Reader is defined over the full vocabulary rather than just the candidate answer set $\mathcal{A}$:

$$P(\cdot|p,q,\mathcal{A}) = \underset{w\in\mathcal{V}}{\text{softmax}}\ e_o(w)^\top \text{MLP}([o, h_q]) \quad (11)$$

Equation (11) is similar to (10) in that it leads to the training of output vectors for the full vocabulary rather than just those items appearing in choice sets in the training data. As in memory networks, this leads to improved performance on non-anonymized datasets.

3.2 Explicit Reference Readers

Attention-Sum Reader. In the Attention-Sum Reader (Kadlec et al., 2016), h and q are computed with equations (5) and (6) as in the Stanford Reader but using GRUs rather than LSTMs. The attention α_t is computed similarly to (7) but using a simple inner product $\alpha_t = \text{softmax}_t\ h_t^\top h_q$ rather than a trained bilinear form. Most significantly, however, equations (8) and (9) are replaced by the following where $t \in R(a,p)$ indicates that a reference to candidate answer a occurs at position t in p.

$$P(a|p,q,\mathcal{A}) = \sum_{t\in R(a,p)} \alpha_t \quad (12)$$

$$\hat{a} = \underset{a}{\text{argmax}} \sum_{t\in R(a,p)} \alpha_t \quad (13)$$

Here we think of $R(a,p)$ as the set of references to a in the passage p. It is important to note that (12) is an equality and that $P(a|p,q,\mathcal{A})$ is not normalized to the members of $R(a,p)$. When training with the log-loss objective this drives the attention α_t to be normalized — to have support only on the positions t with $t \in R(a,p)$ for some a. See the heat maps in the supplementary material.

Gated-Attention Reader. The Gated-Attention Reader (Dhingra et al., 2017) involves a K-layer biGRU architecture defined by the following equations.

$$h_q^\ell = [\text{fGRU}(e(q))_{|q|}, \text{bGRU}(e(q))_1] \;\; 1 \le \ell \le K$$
$$h^1 = \text{biGRU}(e(p))$$
$$h^\ell = \text{biGRU}(h^{\ell-1} \odot h_q^{\ell-1}) \;\; 2 \le \ell \le K$$

Here the question embeddings h_q^ℓ for different values of ℓ are computed with different GRU model parameters. Here $h \odot h_q$ abbreviates the sequence $h_1 \odot h_q, h_2 \odot h_q, \ldots h_{|p|} \odot h_q$. Note that for $K = 1$ we have only h_q^1 and h^1 as in the attention-sum reader. An attention is then computed over the final layer h^K with $\alpha_t = \text{softmax}_t \; (h_t^K)^\top \; h_q^K$ in the Attention-Sum Reader. This reader uses (12) and (13).

Attention-over-Attention Reader. The Attention-over-Attention Reader (Cui et al., 2017) uses a more elaborate method to compute the attention α_t. We will use t to range over positions in the passage and j to range over positions in the question. The model is then defined by the following equations.

$$h = \text{biGRU}(e(p)) \qquad h_q = \text{biGRU}(e(q))$$

$$\alpha_{t,j} = \text{softmax}_t \, h_t^\top h_{q,j} \quad \beta_{t,j} = \text{softmax}_j \, h_t^\top h_{q,j}$$

$$\beta_j = \tfrac{1}{|p|} \textstyle\sum_t \beta_{t,j} \qquad \alpha_t = \textstyle\sum_j \beta_j \alpha_{t,j}$$

Note that the final equation defining α_t can be interpreted as applying the attention β_j to the attentions $\alpha_{t,j}$. This reader uses (12) and (13).

4 Emergent Predication Structure

As discussed in the introduction the entity identifiers such as "entity37" introduced in the CNN/Daily Mail datasets cannot be assigned any semantics other than their identity. We should think of them as pointers or semantics-free constant symbols. Despite this undermining of semantics, aggregation readers using (7) and (9) are able to perform well. Here we posit that this is due to an emergent predication structure in the hidden vectors h_t. Intuitively we want to think of the hidden state vector h_t as a concatenation $[s(\Phi_t), s(a_t)]$ where Φ_t is a property being asserted of entity a_t at the positon t in the passage. Here $s(\Phi_t)$ and $s(a_t)$ are emergent embeddings of the property and entity respectively, We also think of the vector representation q of the question as having the form $[s(\Psi), 0]$ and the vector embedding $e_o(a)$ as having the form $[0, s(a)]$.

Unfortunately, the decomposition of h_t into this predication structure need not be axis aligned. Rather than posit an axis-aligned concatenation we posit that the hidden vector space H is a possibly non-aligned direct sum

$$H = S \oplus E \tag{14}$$

where S is a subspace of "statement vectors" and E is an orthogonal subspace of "entity pointers". Each hidden state vector $h \in H$ then has a unique decomposition as $h = \Psi + e$ for $\Psi \in S$ and $e \in E$. This is equivalent to saying that the hidden vector space H is some rotation of a concatenation of the vector spaces S and E. In this non-axis aligned model we also assume emergent embeddings $s(\Phi)$ and $s(a)$ with $s(\Phi) \in S$ and $s(a) \in E$. We will also assume that the latent spaces are learned in such a way that explicit entity output embeddings satisfy $e_o(a) \in E$.

We now present empirical evidence for this decomposition structure. This structure implies $e_o(a)^\top h_t$ equals $e_o(a)^\top s(a_t)$. This suggests the following for some fixed positive constant c.

$$e_o(a)^\top h_t = \begin{cases} c & \text{if } t \in R(a,p) \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

We note that if $e_o(a)^\top s(a)$ was different for different constant a then answers would be biased toward constant symbols where this product was larger. But we need to have that all constant symbols are equivalent. We note that (15) gives

$$\underset{a}{\text{argmax}} \; e_o(a)^\top o = \underset{a}{\text{argmax}} \; e_o(a)^\top \sum_t \alpha_t h_t$$
$$= \underset{a}{\text{argmax}} \sum_t \alpha_t \, e_o(a)^\top h_t = \underset{a}{\text{argmax}} \sum_{t \in R(a,p)} \alpha_t$$

and hence (9) and (13) agree — the aggregation readers and the explicit reference readers are using essentially the same answer selection criterion.

Empirical evidence for (15) is given in the first three rows of Table 1. The first row empirically measures the constant c in (15) by measuring $e_0(a)^\top h_t$ for those cases where $t \in R(a,p)$. The second row measures "0" in (15) by measuring $e_o(a)^\top h_t$ in those cases where $t \notin R(a,p)$. The third row shows that this inner product falls off significantly just one word before or after the

	CNN Dev			CNN Test		
	samples	mean	variance	samples	mean	variance
$e_o(a)^\top h_t, \quad t \in R(a,p)$	222,001	10.66	2.26	164,746	10.70	2.45
$e_o(a)^\top h_t, \quad t \notin R(a,p)$	93,072,682	-0.57	1.59	68,451,660	-0.58	1.65
$e_o(a)^\top h_{t\pm1}, \quad t \in R(a,p)$	443,878	2.32	1.79	329,366	2.25	1.84
$\text{Cosine}(h_q, h_t), \quad \exists a\ t \in R(a,p)$	222,001	0.22	0.11	164,746	0.22	0.12
$\text{Cosine}(h_q, e_o(a)), \quad \forall a$	103,909	-0.03	0.04	78,411	-0.03	0.04

Table 1: Statistics to support (15) and (16). These statistics are computed for the Stanford Reader.

position of the answer word. Additional evidence for (15) is given in Figure 1 showing that the output vectors $e_o(a)$ for different entity identifiers a are nearly orthogonal. Orthogonality of the output vectors is required by (15) provided that each output vector $e_o(a)$ is in the span of the hidden state vectors $h_{t,p}$ for which $t \in R(a,p)$. Intuitively, the mean of all vectors $h_{t,p}$ with $t \in R(a,p)$ should be approximately equal to $e_o(a)$. Empirically this will only be approximately true.

Equation (15) would suggest that the vector embedding of the constant symbols should have dimension at least as large as the number of distinct constants. However, in practice it is sufficient that $e_0(a)^\top s(a')$ is small for $a \neq a'$. This allows the vector embeddings of the constants to have dimension much smaller than the number of constants. We have experimented with two-sparse constant symbol embeddings where the number of embedding vectors in dimension d is $2d(d-1)$ (d choose 2 times the four ways of setting the signs of the non-zero coordinates). Although we do not report results here, these designed and untrained constant embeddings worked reasonably well.

As further support for (15) we give heat maps for $e_o(a)^\top h_t$ for different identifiers a and heat maps for α_t for different readers in the supplementary material.

As another testable predication we note that the posited decomposition of the hidden state vectors implies

$$h_q^\top (h_i + e_o(a)) = h_q^\top h_i. \tag{16}$$

This equation is equivalent to $h_q^\top e_o(a) = 0$. Experimentally, however, we cannot expect $h_q^\top e_o(a)$ to be exactly zero and (16) seems to provides a more experimentally meaningful test. Empirical evidence for (16) is given in the fourth and fifth rows of Table 1. The fourth row measures the

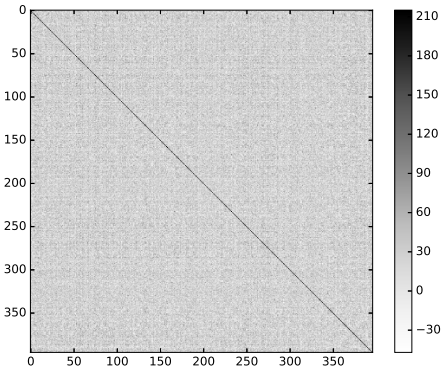

Figure 1: Plot of $e_o(a_i)^\top e_o(a_j)$ from Stanford Reader trained on CNN dataset, where rows range over i values and columns range over j values. Off-diagonal values have mean 25.6 and variance 17.2 while diagonal values have mean 169 and variance 17.3.

cosine of the angle between the question vector h_q and the hidden state h_t averaged over passage positions t at which some entity identifier occurs. The fifth row measures the cosine of the angle between h_q and $e_o(a)$ averaged over the entity identifiers a.

A question asks for a value of x such that a statement $\Psi[x]$ is implied by the passage. For a question Ψ we might even suggest the following vectorial interpretation of entailment.

$$\Phi[x] \text{ implies } \Psi[x] \quad \text{iff} \quad \Phi^\top \Psi \geq ||\Psi||_1.$$

This interpretation is exactly correct if some of the dimensions of the vector space correspond to predicates, Ψ is a 0-1 vector representing a conjunction predicates, and Φ is also 0-1 on these dimensions indicating whether a predicate is implied by the context. Of course in practice one expects the dimension to be smaller than the number of possible predicates.

5 Pointer Annotation Readers

It is of course important to note that anonymization provides reference information—anonymization assumes that one can determine coreference so as to replace coreferent phrases with the same entity identifier. Anonymization allows the reference set $R(a, p)$ to be directly read off of the passage. Still, an aggregation reader must learn to recover this explicit reference structure.

Aggregation readers can have difficulty when anonymization is not done. The Stanford Reader achieves just better than 45% on the Who-did-What dataset while the Attention-Sum Reader can get near 60% (see Table 2). But if we anonymize the Who-did-What dataset and then re-train the Stanford Reader, the accuracy jumps to near 65%. Anonymization greatly reduces the number of output word embeddings $e_o(a)$ to be learned. We need to learn only output embeddings for the relatively small number of entity identifiers needed for the question. Anonymization suppresses the semantics of the reference phrases and leaves only a semantics-free entity identifier. This suppression of semantics may facilitate the separation of the hidden state vector space H into a direct sum $S \oplus E$ with $s(\Phi) \in S$ and $e_o(a), s(a) \in E$.

A third, and perhaps more important effect of anonymization is to provide reference information. Anonymization explicitly marks positions of candidate answers and establishes coreference. A natural question is whether this information can be provided without anonymization by simply adding additional coreference features to the input. Here we evaluate two architectures inspired by this question. This evaluation is done on the Who-did-What dataset which is not anonymized. In each architecture we add features to the input to mark the occurrences of candidate answers. These models are simpler than the Stanford Reader but perform comparably. This comparable performance in Table 2 further supports our analysis of logical structure in aggregation readers.

One-Hot Pointer Reader: The Stanford Reader uses input embeddings of words and output embeddings of entity identifiers. In the Who-did-What dataset each problem has at most five choices in the multiple choice answer list. This means that we need only five entity identifiers and we can use a five dimensional one-hot vector representation for answer identifiers.

If an answer choice exists at position t in the passage let i_t be the index of that choice on the answer choice list. If no answer choice occurs at position t we let i_t be zero. We define $e'(i)$ to be the zero vector if $i = 0$ and otherwise to be the one-hot vector for i (i.e., the five-dimensional vector with zeroes at all positions except with a one at position i). We define "pointer annotation" to be the result of concatenating $e'(i_t)$ as additional features to the word embedding $e(w_t)$ for token w_t in the passage:

$$\bar{e}(w_t) = [e(w_t), e'(i_t)] \tag{17}$$

We feed the new $\bar{e}(w_t)$ to the readers for each token w_t. We define a "one-hot pointer reader" by designating the last five dimensions of the hidden state as indicators of the answer and take the probability of choice i to be defined as

$$p(i|d, q) = \underset{i \in \mathcal{A}}{\text{softmax}}\ o_i \tag{18}$$

where o is computed by (7) and o_i is the ith-to-last dimension of vector o. Table 2 shows results using this reader, showing performance comparable to the Stanford Reader with anonymization.

General Pointer Reader: In the CNN dataset there are roughly 550 entity identifiers and a one-hot representation may not be desirable because it would enlarge the embedding space too much. Instead we can let $e'(i)$ be a fixed set of "pointer vectors"—vectors distributed widely on the unit sphere so that for $i \neq j$ we have that $e'(i)^\top e'(j)$ is small. We again use (17) but replace (18) with

$$p(i|d, q) = \underset{i}{\text{softmax}}\ [0, e'(i)]^\top o \tag{19}$$

where "0" stands for a sufficient number of zeroes in order to make the dimensions match. We refer to this as a "general pointer reader". In this reader, the pointer embeddings $e'(i)$ are held fixed and not trained. Even though not shown here, in preliminary experiments, this reader yield similar performance to the one hot pointer reader while permitting smaller embedding dimensionality.

Linguistic Features: Each model can be modified to include additional input features for each input token in the question and passage. More specifically we can add the following features to the word embeddings: whether the current token occurs in the question; the frequency of the current token in

the passage; the position of the token's first occurrence in the passage as a percentage of the passage length; and whether the text surrounding the token matches the text surrounding the placeholder in the question. More details of the experimental setup are provided in the appendix.

Table 2 shows results when adding these features to the Gated-Attention Reader, Stanford Reader, and One-Hot Pointer Reader, showing large improvements to all readers and leading to the best single-model performance reported to-date on the Who-did-What dataset.

6 Discussion

Explicit reference architectures rely on reference resolution—a specification of which phrases in the given passage refer to candidate answers. Our experiments indicate that all existing readers benefit greatly from this externally provided information. Aggregation readers seem to demonstrate a stronger learning ability in that they essentially learn to mimic explicit reference readers by identifying reference annotation and using it appropriately. This is done most clearly in the pointer reader architectures. Furthermore, we have argued for, and given experimental evidence for, an interpretation of aggregation readers as learning emergent predication structure—a factoring of neural representations into a direct sum of a statement (predicate) representation and an entity (argument) representation.

At a very high level our analysis and experiments support a central role for reference resolution in reading comprehension. Automating reference resolution in neural models, and demonstrating its value on appropriate datasets, would seem to be an important area for future research.

There is great interest in learning representations for natural language understanding. The current state of the art in reading comprehension is such that systems still benefit from externally provided linguistic features including externally annotated reference resolution. It would be interesting to develop fully automated neural readers that perform as well as readers using externally provided annotations.

Acknowledgments

We thank NVIDIA Corporation for donating GPUs used in this research.

References

Frederic Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian J. Goodfellow, Arnaud Bergeron, Nicolas Bouchard, and Yoshua Bengio. 2012. Theano: new features and speed improvements. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*.

Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A thorough examination of the CNN/Daily Mail reading comprehension task. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Zewei Chu, Hai Wang, Kevin Gimpel, and David McAllester. 2017. Broad context language modeling as reading comprehension. In *Proceedings of the 15th Conference of the European Chapter of the ACL (EACL)*.

Yiming Cui, Zhipeng Chen, Si Wei, Shijin Wang, Ting Liu, and Guoping Hu. 2017. Attention-over-attention neural networks for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Graff David and Christopher Cieri. 2003. English Gigaword LDC2003T05. Linguistic Data Consortium.

Bhuwan Dhingra, Hanxiao Liu, Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. 2017. Gated-attention readers for text comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Karl Moritz Hermann, Tomas Kocisky, 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*.

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2016. The Goldilocks principle: Reading children's books with explicit memory representations. In *Proceedings of the 4th International Conference on Learning Representations*.

Pennington Jeffrey, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing (EMNLP)*.

Rudolf Kadlec, Martin Schmid, Ondřej Bajgar, and Jan Kleindienst. 2016. Text understanding with the attention sum reader network. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*.

Who did What	Validation	Test
Attention Sum Reader (Onishi et al., 2016)	59.8	58.8
Gated Attention Reader (Onishi et al., 2016)	60.3	59.6
NSE (Munkhdalai and Yu, 2016)	66.5	66.2
Gated Attention + Linguistic Features$^+$	72.2	**72.8**
Stanford Reader	46.1	45.8
Attentive Reader with Anonymization	55.7	55.5
Stanford Reader with Anonymization	64.8	64.5
One-Hot Pointer Reader	65.1	64.4
One-Hot Pointer Reader + Linguistic Features$^+$	69.3	68.7
Stanford with Anonymization + Linguistic Features$^+$	69.7	**69.2**
Human Performance	-	84

Table 2: Accuracy on Who-did-What dataset. Each result is based on a single model. Results for neural readers other than NSE are based on replications of those systems. All models were trained on the relaxed training set which uniformly yields better performance than the restricted training set. The first group of models are explicit reference models and the second group are aggregation models. + indicates anonymization with better reference identifier.

Moontae Lee, Xiaodong He, Scott Wen tau Yih, Jianfeng Gao, Li Deng, and Paul Smolensky. 2016. Reasoning in vector space: An exploratory study of question answering. In *Proceedings of the 4th International Conference on Learning Representations*.

Tsendsuren Munkhdalai and Hong Yu. 2016. Reasoning with memory augmented neural networks for language comprehension. *arXiv* .

Takeshi Onishi, Hai Wang, Mohit Bansal, Kevin Gimpel, and David McAllester. 2016. Who did What: A large-scale person-centered cloze dataset. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernandez. 2016. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

Pascanu Razvan, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *Proceedings of International Conference on Machine Learning (ICML)*.

Matthew Richardson, Christopher JC Burges, and Erin Renshaw. 2013. MCTest: A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing (EMNLP)*.

Andrew M. Saxe, James L. McClelland, and Surya Ganguli. 2013. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv* .

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In *Advances in Neural Information Processing Systems*.

Bart van Merrienboer, Dzmitry Bahdanau, Vincent Dumoulin, Dmitriy Serdyuk, David Warde-farley, Jan Chorowski, and Yoshua Bengio. 2015. Blocks and fuel: Frameworks for deep learning. *arXiv* .

Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M. Rush, Bart van Merrienboer, Armand Joulin, and Tomas Mikolov. 2016. Towards AI complete question answering: A set of prerequisite toy tasks. In *Proceedings of the 4th International Conference on Learning Representations*.

Jason Weston, Sumit Chopra, and Antoine Bordes. 2015. Memory networks. In *Proceedings of the 3rd International Conference on Learning Representations*.

A Supplemental Material

A.1 Experiment Details

We implemented the neural readers using Theano (Bastien et al., 2012) and Blocks (van Merrienboer et al., 2015) and train them on a single NVIDIA Tesla K40 GPU. Negative log-likelihood is employed as training criterion. We used stochastic gradient descent (SGD) with the Adam update rule (Kingma and Ba, 2015) and set the learning rate to 0.0005.

For the Stanford Reader and One-Hot Pointer Reader, we use the Stanford Reader's default settings. For the Gated-Attention reader, the lookup table was initialized using pre-trained GloVe (Jeffrey et al., 2014) vectors.[1] Input to hidden state weights were initialized by random orthogonal matrices (Saxe et al., 2013) and biases were initialized to zero. Hidden to hidden state weights were initialized by identity matrices to force the model to remember longer information. To compute the attention weight, we use $\alpha_t = h_t^\top W_\alpha h_q$ and initialize W_α with random uniform distribution. We also use gradient clipping (Razvan et al., 2013) with a threshold of 10 and mini-batches of size 32.

During training we randomly shuffle all examples within each epoch. To speed up training, we always pre-fetch 10 batches worth of examples and sort them according to document length as done by Kadlec et al. (2016). When using anonymization, we randomly reshuffle the entity identifier to match the procedure proposed by Hermann et al. (2015).

During training we evaluate the accuracy after each epoch and stop training when the accuracy on the validation set starts decreasing. We tried limiting the vocabulary to the most frequent tokens but did not observe any performance improvement compared with using all distinct tokens as the vocabulary. Since part of our experiments need to check word embedding assignment issues, we finally use all the distinct tokens as vocabulary. To find the optimal embedding and hidden state dimension, we tried several groups of different combinations, and the optimal values were 200 and 384, respectively.

When anonymizing the Who-did-What dataset, we can either use simple string matching to replace answers in the question and passage with entity identifiers, or we can use the Stanford named entity recognizer (NER)[2] to detect named entities and replace the answer named entities in the question and passage with entity identifiers. We found the latter to bring 2% improvement compared with simple string matching.

A.2 Heat Maps for Stanford Reader for Different Answer Candidates

We randomly choose one article from the CNN dataset and show $\mathrm{softmax}(e_o(a)^\top h_t)$ for $t \in [0, |p|]$ for each answer candidate a in Figures 2-6. Red color indicates larger probability and orange indicates smaller probability and the remaining indicates very low probability that can be ignored. From these figures, we can see that our assumption that $e_o(a)$ is used to pick up its occurrence is reasonable.

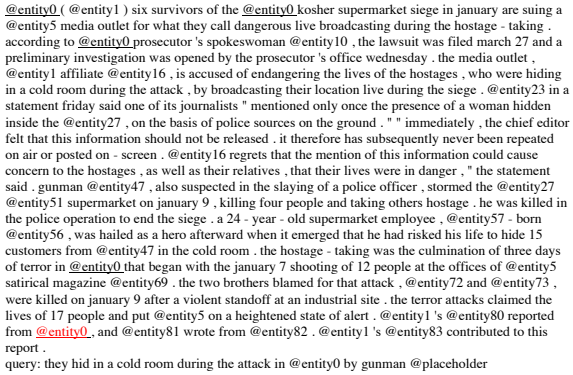

Figure 2: Heat map when a = entity0.

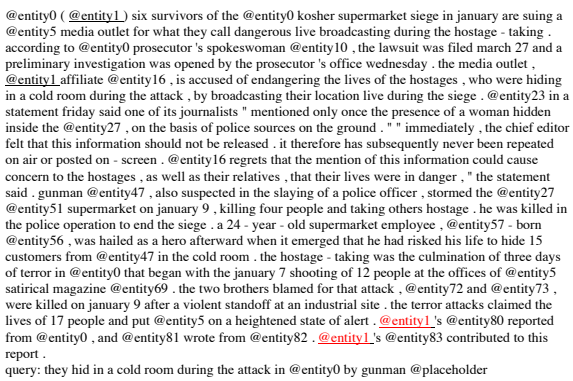

Figure 3: Heat map when a = entity1.

A.3 Heat Maps for Other Readers

We randomly choose one article from the CNN dataset and show the attention map α_t =

[1] http://nlp.stanford.edu/data/glove.6B.zip

[2] http://nlp.stanford.edu/software/CRF-NER.shtml

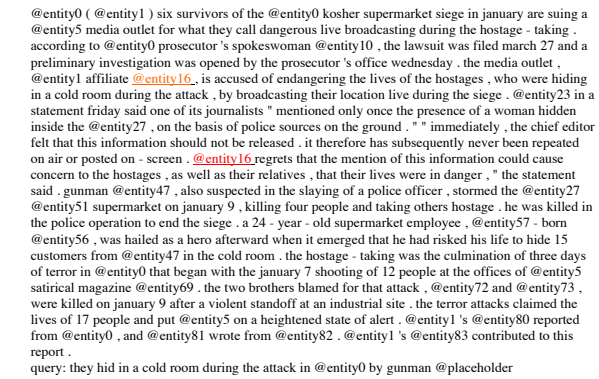

Figure 4: Heat map when a = entity16.

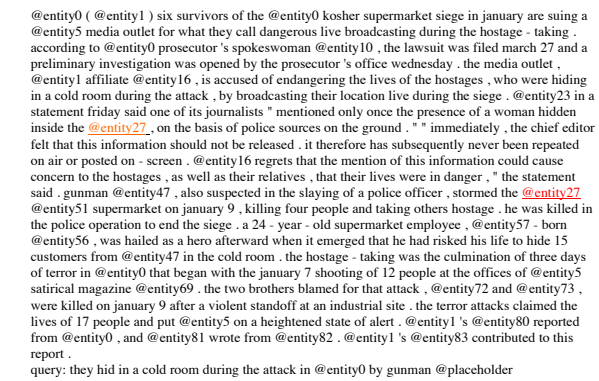

Figure 5: Heat map when a = entity27.

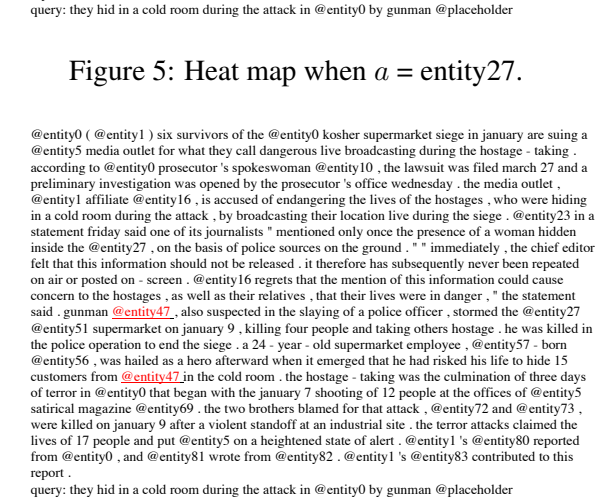

Figure 6: Heat map when a = entity47.

$\text{softmax}(h_q^\top W_a h_t)$ for different readers (in Attention Sum and Gated Attention Reader, W_α is identity matrix). In Figures 7-9, we can see that all readers put essentially all weight on the entity identifiers.

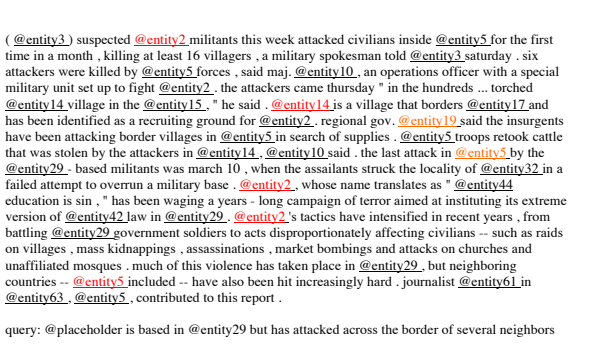

Figure 7: Heat map α_t for Stanford Reader.

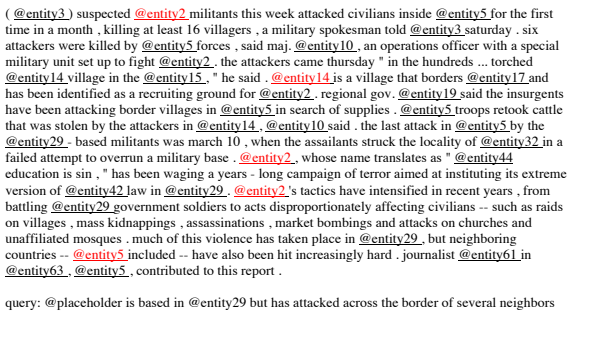

Figure 8: Heat map α_t for Gated Attention Reader.

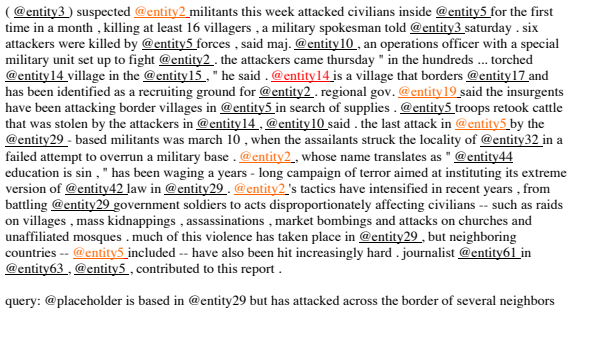

Figure 9: Heat map α_t for Attention Sum Reader.