# A Thorough Examination of the CNN/Daily Mail Reading Comprehension Task

**Danqi Chen** and **Jason Bolton** and **Christopher D. Manning**
Computer Science Stanford University
Stanford, CA 94305-9020, USA
{danqi,jebolton,manning}@cs.stanford.edu



## Abstract

Enabling a computer to understand a document so that it can answer comprehension questions is a central, yet unsolved goal of NLP. A key factor impeding its solution by machine learned systems is the limited availability of human-annotated data. Hermann et al. (2015) seek to solve this problem by creating over a million training examples by pairing *CNN* and *Daily Mail* news articles with their summarized bullet points, and show that a neural network can then be trained to give good performance on this task. In this paper, we conduct a thorough examination of this new reading comprehension task. Our primary aim is to understand what depth of language understanding is required to do well on this task. We approach this from one side by doing a careful hand-analysis of a small subset of the problems and from the other by showing that simple, carefully designed systems can obtain accuracies of 72.4% and 75.8% on these two datasets, exceeding current state-of-the-art results by over 5% and approaching what we believe is the ceiling for performance on this task.[1]

## 1 Introduction

Reading comprehension (RC) is the ability to read text, process it, and understand its meaning.[2] How to endow computers with this capacity has been an elusive challenge and a long-standing goal of Artificial Intelligence (e.g., (Norvig, 1978)). Genuine reading comprehension involves interpretation of the text and making complex inferences. Human reading comprehension is often tested by asking questions that require interpretive understanding of a passage, and the same approach has been suggested for testing computers (Burges, 2013).

In recent years, there have been several strands of work which attempt to collect human-labeled data for this task – in the form of document, question and answer triples – and to learn machine learning models directly from it (Richardson et al., 2013; Berant et al., 2014; Wang et al., 2015). However, these datasets consist of only hundreds of documents, as the labeled examples usually require considerable expertise and neat design, making the annotation process quite expensive. The subsequent scarcity of labeled examples prevents us from training powerful statistical models, such as deep learning models, and would seem to prevent a system from learning complex textual reasoning capacities.

Recently, researchers at *DeepMind* (Hermann et al., 2015) had the appealing, original idea of exploiting the fact that the abundant news articles of *CNN* and *Daily Mail* are accompanied by bullet point summaries in order to heuristically create large-scale supervised training data for the reading comprehension task. Figure 1 gives an example. Their idea is that a bullet point usually summarizes one or several aspects of the article. If the computer understands the content of the article, it should be able to infer the missing entity in the bullet point.

This is a clever way of creating supervised data cheaply and holds promise for making progress on training RC models; however, it is unclear what level of reading comprehension is actually needed to solve this somewhat artificial task and, indeed, what statistical models that do reasonably well on this task have actually learned.

In this paper, our aim is to provide an in-depth and thoughtful analysis of this dataset and what

[1] Our code is available at https://github.com/danqi/rc-cnn-dailymail.

[2] https://en.wikipedia.org/wiki/Reading_comprehension

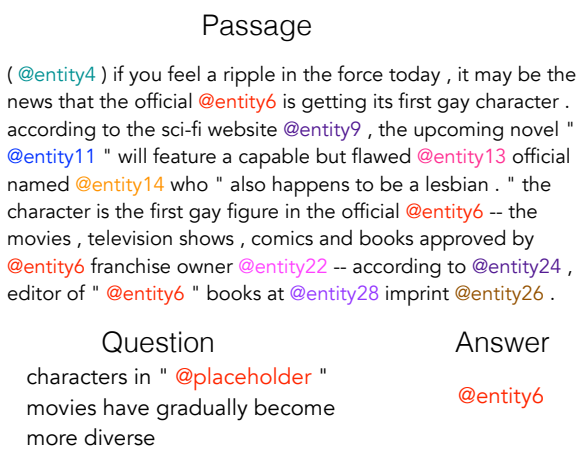


Figure 1: An example item from dataset *CNN*.

level of natural language understanding is needed to do well on it. We demonstrate that simple, carefully designed systems can obtain high, state-of-the-art accuracies of 72.4% and 75.8% on *CNN* and *Daily Mail* respectively. We do a careful hand-analysis of a small subset of the problems to provide data on their difficulty and what kinds of language understanding are needed to be successful and we try to diagnose what is learned by the systems that we have built. We conclude that: (i) this dataset is easier than previously realized, (ii) straightforward, conventional NLP systems can do much better on it than previously suggested, (iii) the distributed representations of deep learning systems are very effective at recognizing paraphrases, (iv) partly because of the nature of the questions, current systems much more have the nature of single-sentence relation extraction systems than larger-discourse-context text understanding systems, (v) the systems that we present here are close to the ceiling of performance for single-sentence and unambiguous cases of this dataset, and (vi) the prospects for getting the final 20% of questions correct appear poor, since most of them involve issues in the data preparation which undermine the chances of answering the question (coreference errors or anonymization of entities making understanding too difficult).

## 2 The Reading Comprehension Task

The RC datasets introduced in (Hermann et al., 2015) are made from articles on the news websites *CNN* and *Daily Mail*, utilizing articles and their bullet point summaries.[3] Figure 1 demonstrates an example[4]: it consists of a passage $p$, a question $q$ and an answer $a$, where the passage is a news article, the question is a cloze-style task, in which one of the article's bullet points has had one entity replaced by a placeholder, and the answer is this questioned entity. The goal is to infer the missing entity (answer $a$) from all the possible entities which appear in the passage. A news article is usually associated with a few (e.g., 3–5) bullet points and each of them highlights one aspect of its content.

The text has been run through a Google NLP pipeline. It it tokenized, lowercased, and named entity recognition and coreference resolution have been run. For each coreference chain containing at least one named entity, all items in the chain are replaced by an @entity$n$ marker, for a distinct index $n$. Hermann et al. (2015) argue convincingly that such a strategy is necessary to ensure that systems approach this task by understanding the passage in front of them, rather than by using world knowledge or a language model to answer questions without needing to understand the passage. However, this also gives the task a somewhat artificial character. On the one hand, systems are greatly helped by entity recognition and coreference having already been performed; on the other, they suffer when either of these modules fail, as they do (in Figure 1, "the character" should probably be coreferent with @entity14; clearer examples of failure appear later on in our data analysis). Moreover, this inability to use world knowledge also makes it much more difficult for a human to do this task – occasionally it is very difficult or impossible for a human to determine the correct answer when presented with an item anonymized in this way.

The creation of the datasets benefits from the sheer volume of news articles available online, so they offer a large and realistic testing ground for statistical models. Table 1 provides some statistics on the two datasets: there are 380k and 879k training examples for *CNN* and *Daily Mail* respectively. The passages are around 30 sentences and 800 tokens on average, while each question contains around 12–14 tokens.

In the following sections, we seek to more deeply understand the nature of this dataset. We first build some straightforward systems in order to get a better idea of a lower-bound for the performance of

[3]The datasets are available at `https://github.com/deepmind/rc-data`.

[4]The original article can be found at `http://www.cnn.com/2015/03/10/entertainment/feat-star-wars-gay-character/`.

| | CNN | Daily Mail |
|---|---|---|
| # Train | 380,298 | 879,450 |
| # Dev | 3,924 | 64,835 |
| # Test | 3,198 | 53,182 |
| Passage: avg. tokens | 761.8 | 813.1 |
| Passage: avg. sentences | 32.3 | 28.9 |
| Question: avg. tokens | 12.5 | 14.3 |
| Avg. # entities | 26.2 | 26.2 |

Table 1: Data statistics of the *CNN* and *Daily Mail* datasets. The avg. tokens and sentences in the passage, the avg. tokens in the query, and the number of entities are based on statistics from the training set, but they are similar on the development and test sets.

current NLP systems. Then we turn to data analysis of a sample of the items to examine their nature and an upper bound on performance.

## 3 Our Systems

In this section, we describe two systems we implemented – a conventional entity-centric classifier and an end-to-end neural network. While Hermann et al. (2015) do provide several baselines for performance on the RC task, we suspect that their baselines are not that strong. They attempt to use a frame-semantic parser, and we feel that the poor coverage of that parser undermines the results, and is not representative of what a straightforward NLP system – based on standard approaches to factoid question answering and relation extraction developed over the last 15 years – can achieve. Indeed, their frame-semantic model is markedly inferior to another baseline they provide, a heuristic word distance model. At present just two papers are available presenting results on this RC task, both presenting neural network approaches: (Hermann et al., 2015) and (Hill et al., 2016). While the latter is wrapped in the language of end-to-end memory networks, it actually presents a fairly simple window-based neural network classifier running on the CNN data. Its success again raises questions about the true nature and complexity of the RC task provided by this dataset, which we seek to clarify by building a simple attention-based neural net classifier.

Given the (passage, question, answer) triple $(p, q, a)$, $p = \{p_1, \ldots, p_m\}$ and $q = \{q_1, \ldots, q_l\}$ are sequences of tokens for the passage and question sentence, with $q$ containing exactly one "@placeholder" token. The goal is to infer the correct entity $a \in p \cap E$ that the placeholder corresponds to, where $E$ is the set of all abstract entity markers. Note that the correct answer entity must appear in the passage $p$.

### 3.1 Entity-Centric Classifier

We first build a conventional feature-based classifier, aiming to explore what features are effective for this task. This is similar in spirit to (Wang et al., 2015), which at present has very competitive performance on the MCTest RC dataset (Richardson et al., 2013). The setup of this system is to design a feature vector $f_{p,q}(e)$ for each candidate entity $e$, and to learn a weight vector $\theta$ such that the correct answer $a$ is expected to rank higher than all other candidate entities:

$$\theta^\intercal f_{p,q}(a) > \theta^\intercal f_{p,q}(e), \forall e \in E \cap p \setminus \{a\} \quad (1)$$

We employ the following feature templates:

1. Whether entity $e$ occurs in the passage.
2. Whether entity $e$ occurs in the question.
3. The frequency of entity $e$ in the passage.
4. The first position of occurence of entity $e$ in the passage.
5. $n$-gram exact match: whether there is an exact match between the text surrounding the placeholder and the text surrounding entity $e$. We have features for all combinations of matching left and/or right one or two words.
6. Word distance: we align the placeholder with each occurrence of entity $e$, and compute the average minimum distance of each non-stop question word from the entity in the passage.
7. Sentence co-occurrence: whether entity $e$ co-occurs with another entity or verb that appears in the question, in some sentence of the passage.
8. Dependency parse match: we dependency parse both the question and all the sentences in the passage, and extract an indicator feature of whether $w \xrightarrow{r}$ @placeholder and $w \xrightarrow{r} e$ are both found; similar features are constructed for @placeholder $\xrightarrow{r} w$ and $e \xrightarrow{r} w$.

### 3.2 End-to-end Neural Network

Our neural network system is based on the *AttentiveReader* model proposed by (Hermann et al., 2015). The framework can be described in the following three steps (see Figure 2):

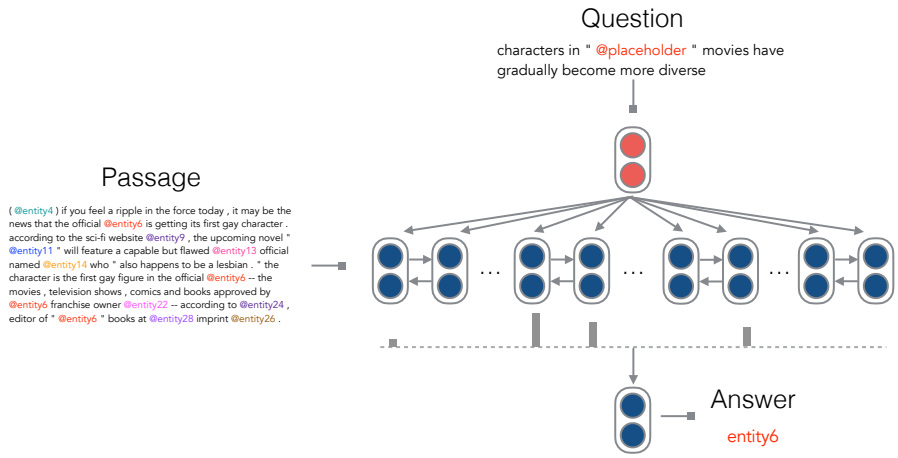


Figure 2: Our neural network architecture for the reading comprehension task.

**Encoding:** First, all the words are mapped to $d$-dimensional vectors via an embedding matrix $E \in \mathbb{R}^{d \times |\mathcal{V}|}$; therefore we have $p$: $\mathbf{p}_1, \ldots, \mathbf{p}_m \in R^d$ and $q : \mathbf{q}_1, \ldots, \mathbf{q}_l \in R^d$.

Next we use a shallow bi-directional LSTM with hidden size $\tilde{h}$ to encode contextual embeddings $\tilde{\mathbf{p}}_i$ of each word in the passage,

$$\begin{aligned} \overrightarrow{\mathbf{h}}_i &= \text{LSTM}(\overrightarrow{\mathbf{h}}_{i-1}, \mathbf{p}_i), i = 1, \ldots, m \\ \overleftarrow{\mathbf{h}}_i &= \text{LSTM}(\overleftarrow{\mathbf{h}}_{i+1}, \mathbf{p}_i), i = m, \ldots, 1 \end{aligned}$$

and $\tilde{\mathbf{p}}_i = \text{concat}(\overrightarrow{\mathbf{h}}_i, \overleftarrow{\mathbf{h}}_i) \in \mathbb{R}^h$, where $h = 2\tilde{h}$. Meanwhile, we use another bi-directional LSTM to map the question $\mathbf{q}_1, \ldots, \mathbf{q}_l$ to an embedding $\mathbf{q} \in \mathbb{R}^h$.

**Attention:** In this step, the goal is to compare the question embedding and all the contextual embeddings, and *select* the pieces of information that are relevant to the question. We compute a probability distribution $\alpha$ depending on the degree of relevance between word $p_i$ (in its context) and the question $q$ and then produce an output vector $\mathbf{o}$ which is a weighted combination of all contextual embeddings $\{\tilde{\mathbf{p}}_i\}$:

$$\alpha_i = \text{softmax}_i \, \mathbf{q}^\intercal \mathbf{W}_s \tilde{\mathbf{p}}_i \quad (2)$$

$$\mathbf{o} = \sum_i \alpha_i \tilde{\mathbf{p}}_i \quad (3)$$

$\mathbf{W_s} \in \mathbb{R}^{h \times h}$ is used in a bilinear term, which allows us to compute a similarity between $\mathbf{q}$ and $\tilde{\mathbf{p}}_i$ more flexibly than with just a dot product.

**Prediction:** Using the *output* vector $\mathbf{o}$, the system outputs the most likely answer using:

$$a = \arg\max_{a \in p \cap E} W_a^\intercal \mathbf{o} \quad (4)$$

Finally, the system adds a softmax function on top of $W_a^\intercal \mathbf{o}$ and adopts a negative log-likelihood objective for training.

**Differences from (Hermann et al., 2015).** Our model basically follows the *AttentiveReader*. However, to our surprise, our experiments observed nearly **8–10**% improvement over the original *AttentiveReader* results on *CNN* and *Daily Mail* datasets (discussed in Sec. 4). Concretely, our model has the following differences:

- We use a bilinear term, instead of a $\tanh$ layer to compute the relevance (attention) between question and contextual embeddings. The effectiveness of the simple bilinear attention function has been shown previously for neural machine translation by (Luong et al., 2015).

- After obtaining the weighted contextual embeddings $\mathbf{o}$, we use $\mathbf{o}$ for direct prediction. In contrast, the original model in (Hermann et al., 2015) combined $\mathbf{o}$ and the question embedding $\mathbf{q}$ via another non-linear layer before making final predictions. We found that we could remove this layer without harming performance. We believe it is sufficient for the model to learn to return the entity to which it maximally gives attention.

- The original model considers all the words from the vocabulary $\mathcal{V}$ in making predictions. We think this is unnecessary, and only predict among entities which appear in the passage.

Of these changes, only the first seems important; the other two just aim at keeping the model simple.

**Window-based MemN2Ns (Hill et al., 2016).** Another recent neural network approach proposed by (Hill et al., 2016) is based on a memory network architecture (Weston et al., 2015). We think it is highly similar in spirit. The biggest difference is their way of encoding passages: they demonstrate that it is most effective to only use a 5-word context window when evaluating a candidate entity and they use a positional unigram approach to encode the contextual embeddings: if a window consists of 5 words $x_1, \ldots, x_5$, then it is encoded as $\sum_{i=1}^{5} E_i(x_i)$, resulting in 5 separate embedding matrices to learn. They encode the 5-word window surrounding the placeholder in a similar way and all other words in the question text are ignored. In addition, they simply use a dot product to compute the "relevance" between the question and a contextual embedding. This simple model nevertheless works well, showing the extent to which this RC task can be done by very local context matching.

## 4 Experiments

### 4.1 Training Details

For training our conventional classifier, we use the implementation of *LambdaMART* (Wu et al., 2010) in the RankLib package.[5] We use this ranking algorithm since our problem is naturally a ranking problem and forests of boosted decision trees have been very successful lately (as seen, e.g., in many recent Kaggle competitions). We do not use all the features of *LambdaMART* since we are only scoring 1/0 loss on the first ranked proposal, rather than using an IR-style metric to score ranked results. We use Stanford's neural network dependency parser (Chen and Manning, 2014) to parse all our document and question text, and all other features can be extracted without additional tools.

For training our neural networks, we only keep the most frequent $|\mathcal{V}| = 50$k words (including entity and placeholder markers), and map all other words to an *¡unk¿* token. We choose word embedding size $d = 100$, and use the 100-dimensional pre-trained *GloVe* word embeddings (Pennington et al., 2014) for initialization. The attention and output parameters are initialized from a uniform distribution between $(-0.01, 0.01)$, and the LSTM weights are initialized from a Gaussian distribution $\mathcal{N}(0, 0.1)$.

[5] https://sourceforge.net/p/lemur/wiki/RankLib/.

| Features | Accuracy |
|---|---|
| Full model | 67.1 |
| − whether $e$ is in the passage | 67.1 |
| − whether $e$ is in the question | 67.0 |
| − frequency of $e$ | **63.7** |
| − position of $e$ | 65.9 |
| − $n$-gram match | **60.5** |
| − word distance | 65.4 |
| − sentence co-occurrence | 66.0 |
| − dependency parse match | 65.6 |

Table 3: Feature ablation analysis of our entity-centric classifier on the development portion of the *CNN* dataset. The numbers denote the accuracy after we exclude each feature from the full system, so a low number indicates an important feature.

We use hidden size $h = 128$ for *CNN* and 256 for *Daily Mail*. Optimization is carried out using vanilla stochastic gradient descent (SGD), with a fixed learning rate of 0.1. We sort all the examples by the length of its passage, and randomly sample a mini-batch of size 32 for each update. We also apply dropout with probability 0.2 to the embedding layer and gradient clipping when the norm of gradients exceeds 10.

All of our models are run on a single GPU (GeForce GTX TITAN X), with roughly a runtime of 6 hours per epoch for *CNN*, and 15 hours per epoch for *Daily Mail*. We run all the models up to 30 epochs and select the model that achieves the best accuracy on the development set.

### 4.2 Main Results

Table 2 presents our main results. The conventional feature-based classifier obtains 67.9% accuracy on the *CNN* test set. Not only does this significantly outperform any of the symbolic approaches reported in (Hermann et al., 2015), it also outperforms all the neural network systems from their paper and the best single-system result reported so far from (Hill et al., 2016). This suggests that the task might not be as difficult as suggested, and a simple feature set can cover many of the cases. Table 3 presents a feature ablation analysis of our entity-centric classifier on the development portion of the *CNN* dataset. It shows that $n$-gram match and frequency of entities are the two most important classes of features.

More dramatically, our single-model neural net-

| Model | CNN | | Daily Mail | |
|---|---|---|---|---|
| | Dev | Test | Dev | Test |
| Frame-semantic model † | 36.3 | 40.2 | 35.5 | 35.5 |
| Word distance model † | 50.5 | 50.9 | 56.4 | 55.5 |
| Deep LSTM Reader † | 55.0 | 57.0 | 63.3 | 62.2 |
| Attentive Reader † | 61.6 | 63.0 | 70.5 | 69.0 |
| Impatient Reader † | 61.8 | 63.8 | 69.0 | 68.0 |
| MemNNs (window memory) ‡ | 58.0 | 60.6 | N/A | N/A |
| MemNNs (window memory + self-sup.) ‡ | 63.4 | 66.8 | N/A | N/A |
| MemNNs (ensemble) ‡ | 66.2* | 69.4* | N/A | N/A |
| Ours: Classifier | 67.1 | 67.9 | 69.1 | 68.3 |
| Ours: Neural net | **72.4** | **72.4** | **76.9** | **75.8** |

Table 2: Accuracy of all models on the *CNN* and *Daily Mail* datasets. Results marked † are from (Hermann et al., 2015) and results marked ‡ are from (Hill et al., 2016). *Classifier* and *Neural net* denote our entity-centric classifier and neural network systems respectively. The numbers marked with * indicate that the results are from ensemble models.

work surpasses the previous results by a large margin (over 5%), pushing up the state-of-the-art accuracies to 72.4% and 75.8% respectively. Due to resource constraints, we have not had a chance to investigate ensembles of models, which generally can bring further gains, as demonstrated in (Hill et al., 2016) and many other papers.

Concurrently with our paper, Kadlec et al. (2016) and Kobayashi et al. (2016) also experiment on these two datasets and report competitive results. However, our single model not only still outperforms theirs, but also appears to be structurally simpler. All these recent efforts converge to similar numbers, and we believe that they are approaching the ceiling performance of this task, as we will indicate in the next section.

## 5 Data Analysis

So far, we have good results via either of our systems. In this section, we aim to conduct an in-depth analysis and answer the following questions: (i) Since the dataset was created in an automatic and heuristic way, how many of the questions are trivial to answer, and how many are noisy and not answerable? (ii) What have these models learned? What are the prospects for further improving them? To study this, we randomly sampled 100 examples from the dev portion of the *CNN* dataset for analysis (see more details in Appendix A).

### 5.1 Breakdown of the Examples

After carefully analyzing these 100 examples, we roughly classify them into the following categories (if an example satisfies more than one category, we classify it into the earliest one):

**Exact match** The nearest words around the placeholder are also found in the passage surrounding an entity marker; the answer is self-evident.

**Sentence-level paraphrasing** The question text is entailed/rephrased by *exactly* one sentence in the passage, so the answer can definitely be identified from that sentence.

**Partial clue** In many cases, even though we cannot find a complete semantic match between the question text and some sentence, we are still able to infer the answer through partial clues, such as some word/concept overlap.

**Multiple sentences** It requires processing multiple sentences to infer the correct answer.

**Coreference errors** It is unavoidable that there are many coreference errors in the dataset. This category includes those examples with critical coreference errors for the answer entity or key entities appearing in the question. Basically we treat this category as "not answerable".

| Category | Question | Passage |
|---|---|---|
| Exact Match | *it 's clear @entity0 is leaning toward* **@placeholder** , says an expert who monitors @entity0 | . . . @entity116 , who follows @entity0 's operations and propaganda closely , recently told @entity3 , *it 's clear @entity0 is leaning toward* **@entity60** in terms of doctrine , ideology and an emphasis on holding territory after operations . . . . |
| Paraphrase | **@placeholder** says he understands why @entity0 wo n't play at his tournament | . . . @entity0 called me personally to let me know that he would n't be playing here at @entity23 , " **@entity3** said on his @entity21 event 's website . . . . |
| Partial clue | a tv movie based on @entity2 's book **@placeholder** casts a @entity76 actor as @entity5 | . . . to @entity12 @entity2 professed that his **@entity11** is not a religious book . . . . |
| Multiple sent. | he 's doing a his - and - her duet all by himself , @entity6 said of **@placeholder** | . . . we got some groundbreaking performances , here too , tonight , @entity6 said . we got **@entity17** , who will be doing some musical performances . he 's doing a his - and - her duet all by himself . . . . |
| Coref. Error | rapper **@placeholder** " disgusted , " cancels upcoming show for @entity280 | . . . with hip - hop star **@entity246** saying on @entity247 that he was canceling an upcoming show for the @entity249 . . . . (but @entity249 = @entity280 = SAEs) |
| Hard | pilot error and snow were reasons stated for **@placeholder** plane crash | . . . a small aircraft carrying **@entity5** , @entity6 and @entity7 the @entity12 @entity3 crashed a few miles from @entity9 , near @entity10 , @entity11 . . . . |

Table 4: Some representative examples from each category.

| No. | Category | (%) |
|---|---|---|
| 1 | Exact match | 13 |
| 2 | Paraphrasing | 41 |
| 3 | Partial clue | 19 |
| 4 | Multiple sentences | 2 |
| 5 | Coreference errors | 8 |
| 6 | Ambiguous / hard | 17 |

Table 5: An estimate of the breakdown of the dataset into classes, based on the analysis of our sampled 100 examples from the *CNN* dataset.

**Ambiguous or very hard** This category includes examples for which we think humans are not able to obtain the correct answer (confidently).

Table 5 provides our estimate of the percentage for each category, and Table 4 presents one representative example from each category. To our surprise, "coreference errors" and "ambiguous/hard" cases account for 25% of this sample set, based on our manual analysis, and this certainly will be a barrier for training models with an accuracy much above 75% (although, of course, a model can sometimes make a lucky guess). Additionally, only 2 examples require multiple sentences for inference – this is a lower rate than we expected and Hermann et al. (2015) suggest. Therefore, we hypothesize that in most of the "answerable" cases, the goal is to identify the most relevant (single) sentence, and then to infer the answer based upon it.

### 5.2 Per-category Performance

Now, we further analyze the predictions of our two systems, based on the above categorization.

As seen in Table 6, we have the following observations: (i) The exact-match cases are quite simple and both systems get 100% correct. (ii) For the ambiguous/hard and entity-linking-error cases, meeting our expectations, both of the systems perform poorly. (iii) The two systems mainly differ in paraphrasing cases, and some of the "partial clue" cases. This clearly shows how neural networks are better capable of learning semantic matches involving paraphrasing or lexical variation between the two sentences. (iv) We believe that the neural-net system already achieves near-optimal performance

| Category | Classifier | Neural net |
|---|---|---|
| Exact match | 13 (100.0%) | 13 (100.0%) |
| Paraphrasing | 32 (78.1%) | 39 (95.1%) |
| Partial clue | 14 (73.7%) | 17 (89.5%) |
| Multiple sentences | 1 (50.0%) | 1 (50.0%) |
| Coreference errors | 4 (50.0%) | 3 (37.5%) |
| Ambiguous / hard | 2 (11.8%) | 1 (5.9%) |
| All | 66 (66.0%) | 74 (74.0%) |

Table 6: The per-category performance of our two systems.

on all the single-sentence and unambiguous cases. There does not seem to be much useful headroom for exploring more sophisticated natural language understanding approaches on this dataset.

## 6 Related Tasks

We briefly survey other tasks related to reading comprehension.

**MCTest** (Richardson et al., 2013) is an open-domain reading comprehension task, in the form of fictional short stories, accompanied by multiple-choice questions. It was carefully created using crowd sourcing, and aims at a 7-year-old reading comprehension level.

On the one hand, this dataset has a high demand on various reasoning capacities: over $50\%$ of the questions require multiple sentences to answer and also the questions come in assorted categories (*what*, *why*, *how*, *whose*, *which*, etc). On the other hand, the full dataset has only 660 paragraphs in total (each paragraph is associated with 4 questions), which renders training statistical models (especially complex ones) very difficult.

Up to now, the best solutions (Sachan et al., 2015; Wang et al., 2015) are still heavily relying on manually curated syntactic/semantic features, with the aid of additional knowledge (e.g., word embeddings, lexical/paragraph databases).

**Children Book Test** (Hill et al., 2016) was developed in a similar spirit to the *CNN*/*Daily Mail* datasets. It takes any consecutive 21 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 (question and answer). The questions are also categorized by the type of the missing word: named entity, common noun, preposition or verb. According to the first study on this dataset (Hill et al., 2016), a language model (an $n$-gram model or a recurrent neural network) with local context is sufficient for predicting verbs or prepositions; however, for named entities or common nouns, it improves performance to scan through the whole paragraph to make predictions. So far, the best published results are reported by window-based memory networks.

**bAbI** (Weston et al., 2016) is a collection of artificial datasets, consisting of 20 different reasoning types. It encourages the development of models with the ability to chain reasoning, induction/deduction, etc., so that they can answer a question like "The football is in the *playground*" after reading a sequence of sentences "John is in the playground; Bob is in the office; John picked up the football; Bob went to the kitchen." Various types of memory networks (Sukhbaatar et al., 2015; Kumar et al., 2016) have been shown effective on these tasks, and Lee et al. (2016) show that vector space models based on extensive problem analysis can obtain near-perfect accuracies on all the categories. Despite these promising results, this dataset is limited to a small vocabulary (only 100–200 words) and simple language variations, so there is still a huge gap from real-world datasets that we need to fill in.

## 7 Conclusion

In this paper, we carefully examined the recent *CNN*/*Daily Mail* reading comprehension task. Our systems demonstrated state-of-the-art results, but more importantly, we performed a careful analysis of the dataset by hand.

Overall, we think the *CNN*/*Daily Mail* datasets are valuable datasets, which provide a promising avenue for training effective statistical models for reading comprehension tasks. Nevertheless, we argue that: (i) this dataset is still quite noisy due to its method of data creation and coreference errors; (ii) current neural networks have almost reached a performance ceiling on this dataset; and (iii) the required reasoning and inference level of this dataset is still quite simple.

As future work, we need to consider how we can utilize these datasets (and the models trained upon them) to help solve more complex RC reasoning tasks (with less annotated data).

## Acknowledgments

We thank the anonymous reviewers for their thoughtful feedback. Stanford University gratefully

acknowledges the support of the Defense Advanced Research Projects Agency (DARPA) Deep Exploration and Filtering of Text (DEFT) Program under Air Force Research Laboratory (AFRL) contract no. FA8750-13-2-0040. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of the DARPA, AFRL, or the US government.

## References

Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang, Peter Clark, and Christopher D. Manning. 2014. Modeling biological processes for reading comprehension. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1499–1510.

Christopher J.C. Burges. 2013. Towards the machine comprehension of text: An essay. Technical report, Microsoft Research Technical Report MSR-TR-2013-125.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750.

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 (NIPS)*, pages 1684–1692.

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2016. The goldilocks principle: Reading children's books with explicit memory representations. In *International Conference on Learning Representations (ICLR)*.

Rudolf Kadlec, Martin Schmid, Ondrej Bajgar, and Jan Kleindienst. 2016. Text understanding with the attention sum reader network. In *Association for Computational Linguistics (ACL)*.

Sosuke Kobayashi, Ran Tian, Naoaki Okazaki, and Kentaro Inui. 2016. Dynamic entity representation with max-pooling improves machine reading. In *North American Association for Computational Linguistics (NAACL)*.

Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. 2016. Ask me anything: Dynamic memory networks for natural language processing. In *International Conference on Machine Learning (ICML)*.

Moontae Lee, Xiaodong He, Wen-tau Yih, Jianfeng Gao, Li Deng, and Paul Smolensky. 2016. Reasoning in vector space: An exploratory study of question answering. In *International Conference on Learning Representations (ICLR)*.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1412–1421.

Peter Norvig. 1978. *A Unified Theory of Inference for Text Understanding*. Ph.D. thesis, University of California, Berkeley.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew Richardson, Christopher J.C. Burges, and Erin Renshaw. 2013. MCTest: A challenge dataset for the open-domain machine comprehension of text. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 193–203.

Mrinmaya Sachan, Kumar Dubey, Eric Xing, and Matthew Richardson. 2015. Learning answer-entailing structures for machine comprehension. In *Association for Computational Linguistics and International Joint Conference on Natural Language Processing (ACL/IJCNLP)*, pages 239–249.

Sainbayar Sukhbaatar, arthur szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2431–2439.

Hai Wang, Mohit Bansal, Kevin Gimpel, and David McAllester. 2015. Machine comprehension with syntax, frames, and semantics. In *Association for Computational Linguistics and International Joint Conference on Natural Language Processing (ACL/IJCNLP)*, pages 700–706.

Jason Weston, Sumit Chopra, and Antoine Bordes. 2015. Memory networks. In *International Conference on Learning Representations (ICLR)*.

Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. 2016. Towards AI-complete question answering: A set of prerequisite toy tasks. In *International Conference on Learning Representations (ICLR)*.

Qiang Wu, Christopher J. Burges, Krysta M. Svore, and Jianfeng Gao. 2010. Adapting boosting for information retrieval measures. *Information Retrieval*, pages 254–270.

## A Samples and Labeled Categories from the *CNN* Dataset

For the analysis in Section 5, we uniformly sampled 100 examples from the development set of the *CNN* dataset. Table 8 provides a full index list of our samples and Table 7 presents our labeled categories.

| Category | Sample IDs |
|---|---|
| Exact match (13) | 8, 11, 23, 27, 28, 32, 43, 57, 63, 72, 86, 87, 99 |
| Sentence-level paraphrasing (41) | 0, 2, 7, 9, 12, 14, 16, 18, 19, 20, 29, 30, 31, 34, 36, 37, 39, 41, 42, 44, 47, 48, 52, 54, 58, 64, 65, 66, 69, 73, 74, 78, 80, 81, 82, 84, 85, 90, 92, 95, 96 |
| Partial clues (19) | 4, 17, 21, 24, 35, 38, 45, 53, 55, 56, 61, 62, 75, 83, 88, 89, 91, 97, 98 |
| Multiple sentences (2) | 5, 76 |
| Coreference errors (8) | 6, 22, 40, 46, 51, 60, 68, 94 |
| Ambiguous or very hard (17) | 1, 3, 10, 13, 15, 25, 26, 33, 49, 50, 59, 67, 70, 71, 77, 79, 93 |

Table 7: Our labeled categories of the 100 samples.

| ID | Filename | ID | Filename |
|---|---|---|---|
| 0 | ddb1e746f88a22fee654ecde8f018e7586595045.question | 1 | 2bef8ec21b10a3294b1496d9a86f29f0592d2300.question |
| 2 | 38c702812a874f983e9890c32ba832841a327351.question | 3 | 636857045cf266dd69b67b1e53617bed5253dc33.question |
| 4 | 417cbffd5e6275b3c42cb88be222a9f6c7d415f1.question | 5 | ef96409c707a699e4055a1d0684eecdb6e115c16.question |
| 6 | b4e157a6a34bf11a03e0b5cd55065c0f39ac8d60.question | 7 | 1d75e7c59978c7c06f3aecaf52bc35b8919eee17.question |
| 8 | 223c8e3aeddc3f65fee1964df17bb72f89b723e4.question | 9 | 13d33b8c86375b0f5fdc856116e91a7355c6fc5a.question |
| 10 | 378fd418b8ec18dff406be07ec225e6bf53659f5.question | 11 | d8253b7f22662911c19ec4468f81b9db29df1746.question |
| 12 | 80529c792d3a368861b404c1ce4d7ad3c12e552a.question | 13 | 728e7b365e941d814676168c78c9c4f38892a550.question |
| 14 | 3cf6fb2c0d09927a12add82b4a3f248da740d0de.question | 15 | 04b827f84e60659258e19806afe9f8d10b764db1.question |
| 16 | f0abf359d71f7896abd09ff7b3319c70f2ded81e.question | 17 | b6696e0f2166a75fcefbe4f28d0ad06e420eef23.question |
| 18 | 881ab3139c34e9d9f29eb11601321a234d096272.question | 19 | 66f5208d62b543ee41accb7a560d63ff40413bac.question |
| 20 | f83a70d469fa667f0952959346b496fbf3cdb35c.question | 21 | 1853813a80f83a1661dd3f6695559674c749525e.question |
| 22 | 02664d5e3af321afbaf4ee351ba1f24643746451.question | 23 | 20417b5efb836530846ddf677d1bd0bbc831643c.question |
| 24 | 42c25a01801228a863c508f9d9e95399ea5f37a4.question | 25 | 70a3ba822770abcaf64dd131c85ec964d172c312.question |
| 26 | b6636e525ad58ffdc9a7c18187fb3412660d2cdd.question | 27 | 6147c9f2b3d1cc6fbc57c2137f0356513f49bf46.question |
| 28 | 262b855e2f24e1b2e4e0ba01ace81a1f214d729e.question | 29 | d7211f4d21f40461bb59954e53360eeb4bb6c664.question |
| 30 | be813e58ae9387a9fdaf771656c8e1122794e515.question | 31 | ad39c5217042f36e4c1458e9397b4a588bbf8cf9.question |
| 32 | 9534c3907f1cd917d24a9e4f2afc5b38b82d9fca.question | 33 | 3fbe4bfb721a6e1aa60502089c46240d5c332c05.question |
| 34 | 6efa2d6bad587bde65ca22d10eca83cf0176d84f.question | 35 | 436aa25e28d3a026c4fcd658a852b6a24fc6935e.question |
| 36 | 0c44d6ef109d33543cfbd26c95c9c3f6fe33a995.question | 37 | 8472b859c5a8d18454644d9acdb5edd1db175eb5.question |
| 38 | fb4dd20e0f464423b6407fd0d21cc4384905cf26.question | 39 | a192ddbcecf2b00260ae4c7c3c20df4d5ce47a85.question |
| 40 | f7133f844967483519dbf632e2f3fb90c5625a4c.question | 41 | 29b274958eb057e8f1688f02ef8dbc1c6d06c954.question |
| 42 | 8ea6ad57c1c5eb1950f50ea47231a5b3f32dd639.question | 43 | 1e43f2349b17dac6d1b3143f8c5556e2257be92c.question |
| 44 | 7f11f0b4f6bb9aaa3bdc74bffaed5c869b26be97.question | 45 | 8e6d8d984e51adb5071aad22680419854185eaea.question |
| 46 | 57fc2b7ffcfbd1068fbc33b95d5786e2bff24698.question | 47 | 57b773478955811a8077c98840d85af03e1b4f05.question |
| 48 | d857700721b5835c3472ba73ef7abfad0c9c499f.question | 49 | f8eedded53c96e0cb98e2e95623714d2737f29da.question |
| 50 | 4c488f41622ad48977a60c2283910f15a736417e.question | 51 | 39680fd0bff53f2ca02f632eabbc024d698f979e.question |
| 52 | addd9cebe24c96b4a3c8e9a50cd2a57905b6defb.question | 53 | 50317f7a626e23628e4bfd190e987ad5af7d283e.question |
| 54 | 3f7ac912a75e4ef7a56987bff37440ffa14770c6.question | 55 | 610012ef561027623f4b4e3b8310c1c41dc819cc.question |
| 56 | d9c2e9bfc71045be2ecd9596760165 99e4637ed1.question | 57 | 848c068db210e0b255f83c4f8b01d2d421fb9c94.question |
| 58 | f5c2753703b66d26f43bafe7f157803dc96eedbc.question | 59 | 4f76379f1c7b1d4acc5a4c82ced64af6313698dd.question |
| 60 | e5bb1c27d07f1591929bf0283075ad1bc1fc0b50.question | 61 | 33b911f9074c80eb18a57f657ad01393582059be.question |
| 62 | 58c4c046654af52a3cb8f6890411a41c0dd0063b.question | 63 | 7b03f730fda1b247e9f124b692e3298859785ef3.question |
| 64 | ece6f4e047856d5a84811a67ac9780d48044e69a.question | 65 | 35565dc6aecc0f1203842ef13aede0a14a8cf075.question |
| 66 | ddf3f2b06353fe8a9b50043f926eb3ab318e91b2.question | 67 | e248e59739c9c013a2b1b7385d881e0f879b341d.question |
| 68 | e86d3fa2a74625620bcae0003dfbe13416ee29cf.question | 69 | 176bf03c9c19951a8ae5197505a568454a6d4526.question |
| 70 | ee694cb968ae99aea36f910355bf73da417274c0.question | 71 | 7a666f78590edbaf7c4d73c4ea641c545295a513.question |
| 72 | 91e3cdd46a70d6dfbe917c6241eab907da4b1562.question | 73 | e54d9bdcb478ecc490608459d3405571979ef3f2.question |
| 74 | f3737e4de9864f083d6697293be650e02505768c.question | 75 | 1fc7488755d24696a4ed1aabc0a21b8b9755d8c6.question |
| 76 | fb3eadd07b9f1df1f8a7a6b136ad6d06f4981442.question | 77 | 1406bdad74b3f932342718d5d5d0946a906d73e2.question |
| 78 | 54b6396669bdb2e30715085745d4f98d058269ef.question | 79 | 0a53102673f2bebc36ce74bf71db1b42a0187052.question |
| 80 | d5eb4f98551d23810bfeb0e5b8a94037bcf58b0d.question | 81 | 370de4ffe0f2f9691e4bd456ff344a6a337e0edf.question |
| 82 | 12f32c770c86083ff21b25de7626505c06440018.question | 83 | 9f6b5cff3ce146e21e323a1462c3eff8fca3d4a0.question |
| 84 | 1c2a14f525fa3802b8da52aebaa9abd2091f9215.question | 85 | f2416e14d89d40562284ba2d15f7d5cc59c7e602.question |
| 86 | adcf5881856bcbaf1ad93d06a3c5431f6a0319ba.question | 87 | 097d34b804c4c052591984d51444c4a97a3c41ac.question |
| 88 | 773066c39bb3b593f676caf03f7e7370a8cd2a43.question | 89 | 598cf5ff08ea75dcedda31ac1300e49cdf90893a.question |
| 90 | b66ebaaefb844f1216fd3d28eb160b08f42cde62.question | 91 | 535a44842decdc23c11bae50d9393b923897187e.question |
| 92 | e27ca3104a596171940db8501c4868ed2fbc8cea.question | 93 | bb07799b4193cffa90792f92a8c14d591754a7f3.question |
| 94 | 83ff109c6ccd512abdf317220337b98ef551d94a.question | 95 | 5ede07a1e4ac56a0155d852df0f5bb6bde3cb507.question |
| 96 | 7a2a9a7fbb44b0e51512c61502ce2292170400c1.question | 97 | 9dcdc052682b041cdbf2fadc8e55f1bafc88fe61.question |
| 98 | 0c2e28b7f373f29f3796d29047556766cc1dd709.question | 99 | 2bdf1696bfd2579bb719402e9a6fa99cb8dbf587.question |

Table 8: A full index list of our samples.