A Simple Method for Commonsense Reasoning

Trieu H. Trinh[*]
Google Brain
thtrieu@google.com

Quoc V. Le
Google Brain
qvl@google.com

Abstract

Commonsense reasoning is a long-standing challenge for deep learning. For example, it is difficult to use neural networks to tackle the Winograd Schema dataset [1]. In this paper, we present a simple method for commonsense reasoning with neural networks, using unsupervised learning. Key to our method is the use of language models, trained on a massive amount of unlabled data, to score multiple choice questions posed by commonsense reasoning tests. On both Pronoun Disambiguation and Winograd Schema challenges, our models outperform previous state-of-the-art methods by a large margin, without using expensive annotated knowledge bases or hand-engineered features. We train an array of large RNN language models that operate at word or character level on LM-1-Billion, CommonCrawl, SQuAD, Gutenberg Books, and a customized corpus for this task and show that diversity of training data plays an important role in test performance. Further analysis also shows that our system successfully discovers important features of the context that decide the correct answer, indicating a good grasp of commonsense knowledge.

1 Introduction

Although deep neural networks have achieved remarkable successes (e.g., [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]), their dependence on supervised learning has been challenged as a significant weakness. This dependence prevents deep neural networks from being applied to problems where labeled data is scarce. An example of such problems is common sense reasoning, such as the Winograd Schema Challenge [1], where the labeled set is typically very small, on the order of hundreds of examples. Below is an example question from this dataset:

- *The trophy doesn't fit in the suitcase because **it** is too big. What is too big?*
  ***Answer 0**: the trophy. **Answer 1**: the suitcase*

Although it is straightforward for us to choose the answer to be *"the trophy"* according to our common sense, answering this type of question is a great challenge for machines because there is no training data, or very little of it.

In this paper, we present a surprisingly simple method for common sense reasoning with Winograd schema multiple choice questions. Key to our method is th e use of language models (LMs), trained on a large amount of unlabeled data, to score multiple choice questions posed by the challenge and similar datasets. More concretely, in the above example, we will first substitute the pronoun (*"it"*) with the candidates (*"the trophy"* and *"the suitcase"*), and then use LMs to compute the probability of the two resulting sentences (*"The trophy doesn't fit in the suitcase because **the trophy** is too big."* and *"The trophy doesn't fit in the suitcase because **the suitcase** is too big."*). The substitution that results in a more probable sentence will be the correct answer.

[*]Work done as a member of the Google Brain Residency program (g.co/brainresidency.)

Preprint. Work in progress.

On both Pronoun Disambiguation and Winograd Schema challenges, our method outperforms previous state-of-the-art methods by a large margin, without using expensive annotated knowledge bases or hand-engineered features. On a Pronoun Disambiguation dataset, PDP-60, our method achieves 70.0% accuracy, which is better than the state-of-art accuracy of 66.7%. On a Winograd Schema dataset, WSC-273, our method achieves 63.7% accuracy, 11% above that of the current state-of-art result (52.8%)[2]

A unique feature of Winograd Schema questions is the presence of a special word that decides the correct reference choice. In the above example, *"big"* is this special word. When *"big"* is replaced by *"small"*, the correct answer switches to *"the suitcase"*. Although detecting this feature is not part of the challenge, further analysis shows that our system successfully discovers this special word to make its decisions in many cases, indicating a good grasp of commonsense knowledge.

2 Related Work

Unsupervised learning has been used to discover simple commonsense relationships. For example, Mikolov et al. [16, 17] show that by learning to predict adjacent words in a sentence, word vectors can be used to answer analogy questions such as: Man:King::Woman:?. Our work uses a similar intuition that language modeling can naturally capture common sense knowledge. The difference is that Winograd Schema questions require more contextual information, hence our use of LMs instead of just word vectors.

Neural LMs have also been applied successfully to improve downstream applications [18, 19, 20, 21]. In [18, 19, 20, 21], researchers have shown that pre-trained LMs can be used as feature representations for a sentence, or a paragraph to improve NLP applications such as document classification, machine translation, question answering, etc. The combined evidence suggests that LMs trained on a massive amount of unlabeled data can capture many aspects of natural language and the world's knowledge, especially commonsense information.

Previous attempts on solving the Winograd Schema Challenge usually involve heavy utilization of annotated knowledge bases, rule-based reasoning, or hand-crafted features [22, 23, 24]. In particular, Rahman and Ng [25] employ human annotators to build more supervised training data. Their model utilizes nearly 70K hand-crafted features, including querying data from Google Search API. Sharma et al. [26] rely on a semantic parser to understand the question, query texts through Google Search, and reason on the graph produced by the parser. Similarly, Schüller [24] formalizes the knowledge-graph data structure and a reasoning process based on cognitive linguistics theories. Bailey et al. [23] introduces a framework for reasoning, using expensive annotated knowledge bases as axioms.

The current best approach makes use of the skip-gram model to learn word representations [27]. The model incorporates several knowledge bases to regularize its training process, resulting in Knowledge Enhanced Embeddings (KEE). A semantic similarity scorer and a deep neural network classifier are then combined on top of KEE to predict the answers. The final system, therefore, includes both supervised and unsupervised models, besides three different knowledge bases. In contrast, our unsupervised method is simpler while having significantly higher accuracy. Unsupervised training is done on text corpora which can be cheaply curated.

Using language models in reading comprehension tests also produced many great successes. Namely Chu et al. [28] used bi-directional RNNs to predict the last word of a passage in the LAMBADA challenge. Similarly, LMs are also used to produce features for a classifier in the Store Close Test 2017, giving best accuracy against other methods [29]. In a broader context, LMs are used to produce good word embeddings, significantly improved a wide variety of downstream tasks, including the general problem of question answering [20, 30].

3 Methods

We first substitute the pronoun in the original sentence with each of the candidate choices. The problem of coreference resolution then reduces to identifying which substitution results in a more probable sentence. By reframing the problem this way, language modeling becomes a natural solution

[2]Code to reproduce these results are available at `https://github.com/tensorflow/models/tree/master/research/lm_commonsense`.

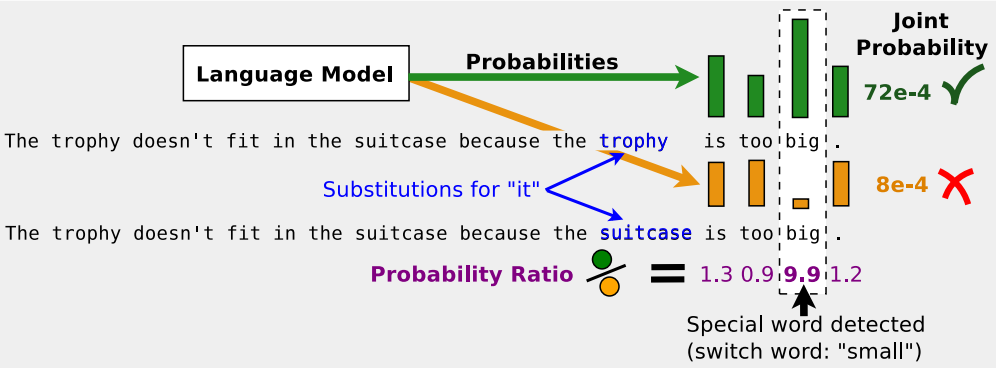


Figure 1: Overview of our method and analysis. We consider the test *"The trophy doesn't fit in the suitcase because* ***it*** *is too big."* Our method first substitutes two candidate references *trophy* and *suitcase* into the pronoun position. We then use an LM to score the resulting two substitutions. By looking at probability ratio at every word position, we are able to detect *"big"* as the main contributor to *trophy* being the chosen answer. When *"big"* is switched to *"small"*, the answer changes to *suitcase*. This switching behaviour is an important feature characterizing the Winograd Schema Challenge.

by its definition. Namely, LMs are trained on text corpora, which encodes human knowledge in the form of natural language. During inference, LMs are able to assign probability to any given text based on what they have learned from training data. An overview of our method is shown in Figure 1.

Suppose the sentence S of n consecutive words has its pronoun to be resolved specified at the k^{th} position: $S = \{w_1, .., w_{k-1}, w_k \equiv p, w_{k+1}, .., w_n\}$. We make use of a trained language model $P_\theta(w_t|w_1, w_2, .., w_{t-1})$, which defines the probability of word w_t conditioned on the previous words $w_1, ..., w_{t-1}$. The substitution of a candidate reference c in to the pronoun position k results in a new sentence $S_{w_k \leftarrow c}$ (we use notation $w_k \leftarrow c$ to mean that word w_k is substituted by candidate c). We consider two different ways of scoring the substitution:

- $Score_{full}(w_k \leftarrow c) = P_\theta(w_1, w_2, ..., w_{k-1}, c, w_{k+1}, ..., w_n)$

which scores how probable the resulting full sentence is, and

- $Score_{partial}(w_k \leftarrow c) = P_\theta(w_{k+1}, ..., w_n|w_1, ..., w_{k-1}, c)$

which scores how probable the part of the resulting sentence following c is, given its antecedent. In other words, it only scores a part $S_{w_k \leftarrow c}$ conditioned on the rest of the substituted sentence. An example of these two scores is shown in Table 1. In our experiments, we find that *partial* scoring strategy is generally better than the naive *full* scoring strategy.

Table 1: Example of *full* and *partial* scoring for the test *"The trophy doesn't fit in the suitcase because* ***it*** *is too big."* with two reference choices *"the suitcase"* and *"the trophy"*.

$c =$ **the suitcase**	$Score_{full}(w_k \leftarrow$ "$the\ suitcase$"$) = P($The trophy doesn't fit in the suitcase because **the suitcase** is too big$)$ $Score_{partial}(w_k \leftarrow$ "$the\ suitcase$"$) = P($is too big$|$ The trophy doesn't fit in the suitcase because **the suitcase**$)$
$c =$ **the trophy**	$Score_{full}(w_k \leftarrow$ "$the\ trophy$"$) = P($The trophy doesn't fit in the suitcase because **the trophy** is too big$)$ $Score_{partial}(w_k \leftarrow$ "$the\ trophy$"$) = P($is too big$|$ The trophy doesn't fit in suitcase because **the trophy**$)$

4 Experimental settings

In this section we describe tests for commonsense reasoning and the LMs used to solve these tasks. We also detail training text corpora used in our experiments.

Evaluation on Commonsense Reasoning Tests. We conduct experiments to evaluate our methods on two tasks: Pronoun Disambiguation Problems and Winograd Schema Challenge. These two tasks

have been proposed as potential alternatives to the Turing Test, specifically targeting its potential weaknesses and inadequacy [1].

On the former task, we use the original set of 60 questions (PDP-60) as the main benchmark[3]. Later analysis augments this test with 62 questions from the development set to avoid bias presented in the original smaller set.[4] The second task (WSC-273) is qualitatively much more difficult[5]. Its recent best reported result is only 3% of accuracy above random guess [27]. This task consists of 273 questions and is designed to work against techniques such as traditional linguistic restrictions, common heuristics or simple statistical test over text corpora ("*Google-proof*") [1].

Recurrent language models. We consider two types of recurrent LMs, one processes word inputs and the other processes character inputs. Their output layer, however, is constructed to only produce word outputs, allowing both types of input processing to join in ensembles. Namely, our LMs predict a distribution over a large vocabulary (800K words) at each time step, using a softmax layer. Following [31], we employ importance sampling at the softmax layer with 8,192 negative samples for each mini-batch to significantly speed up training. We use two layers of LSTM [32] with 8,192 hidden units and a projection layer to a smaller dimensionality at output gates for faster processing.

For models that process words, we use a big embedding look up matrix with vocabulary size 800K and embedding size 1,024. For character-level input, we use a vocabulary size of 256 characters and embedding size 16. Characters in the same word are concatenated and used as input at a single time step. The resulting character embedding is processed using eight convolutions before going into the LSTM layers. More details about our LMs can be found in Appendix A.

Training text corpora. We perform experiments on several different text copora to examine the effect of training data type on test accuracy. Namely, we consider LM-1-Billion, CommonCrawl[6], SQuAD and Gutenberg Books. For SQuAD, we collect context passages from the Stanford Question-Answering Dataset [33] to form its training and validation sets accordingly.

5 Main results

Our experiments start with testing LMs trained on all text corpora with PDP-60 and WSC-273. Next, we show that it is possible to customize training data to obtain even better results.

5.1 The first challenge in 2016: PDP-60

We first examine unsupervised single-model resolvers on PDP-60 by training one character-level and one word-level LM on the Gutenberg corpus. In Table 2, these two resolvers outperform previous results by a large margin. For this task, we found *full* scoring gives better results than *partial* scoring. In Section 6.2, we provide evidences that this is an atypical case due to the very small size of PDP-60.

Table 2: Unsupervised single-model resolver performance on PDP-60

Method	Accuracy
Unsupervised Semantic Similarity Method (USSM)	48.3 %
USSM + Cause-Effect Knowledge Base [34]	55.0 %
USSM + Cause-Effect + WordNet [35] + ConceptNet [36] Knowledge Bases	56.7 %
Char-LM-*partial*	45.0 %
Char-LM-*full*	53.3 %
Word-LM-*partial*	53.3 %
Word-LM-*full*	**60.0 %**

Next, we allow systems to take in necessary components to maximize their test performance. This includes making use of supervised training data that maps commonsense reasoning questions to their

[3] `https://cs.nyu.edu/faculty/davise/papers/WinogradSchemas/PDPChallenge2016.xml`

[4] `http://commonsensereasoning.org/disambiguation.html`

[5] `https://cs.nyu.edu/faculty/davise/papers/WinogradSchemas/WSCollection.xml`

[6] We evaluate all models trained on CommonCrawl after approximately 10-billion words are consumed.

Table 3: Unconstrained resolvers performance on PDP-60

Method	Accuracy
Patric Dhondt (WS Challenge 2016)	45.0 %
Nicos Issak (WS Challenge 2016)	48.3 %
Quan Liu (WS Challenge 2016 - **winner**)	58.3 %
USSM + Supervised Deepnet	53.3 %
USSM + Supervised Deepnet + 3 Knowledge Bases	66.7 %
Ensemble of 5 Unsupervised LMs-*full*	**70.0 %**

correct answer. Here we simply train another three variants of LMs on LM-1-Billion, CommonCrawl, and SQuAD and ensemble all of them. As reported in Table 3, this ensemble of five unsupervised models outperform the best system in the 2016 competition (58.3%) by a large margin. Specifically, we achieve 70.0% accuracy, better than the more recent reported results from Quan Liu et al (66.7%) [27], who makes use of three knowledge bases and a supervised deep neural network.

5.2 Winograd Schema Challenge

On the harder task WSC-273, our single-model resolvers also outperform the current state-of-the-art by a large margin, as shown in Table 4. Namely, our word-level resolver achieves an accuracy of 56.4%. By training another 4 LMs, each on one of the 4 text corpora LM-1-Billion, CommonCrawl, SQuAD, Gutenberg Books, and add to the previous ensemble, we are able to reach 61.5%, nearly 10% of accuracy above the previous best result. This is a drastic improvement considering this previous best system outperforms random guess by only 3% in accuracy.

Table 4: Accuracy on Winograd Schema Challenge

Method	Accuracy
Random guess	50.0%
USSM + Knowledge Base	52.0 %
USSM + Supervised DeepNet + Knowledge Base	52.8 %
Char-LM-*partial*	51.3%
Char-LM-*full*	51.3%
Word-LM-*partial*	56.4%
Word-LM-*full*	53.8%
Ensemble of 10 Unsupervised LMs-*partial*	**61.5 %**

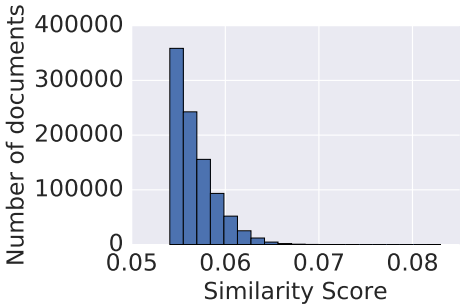
This task is more difficult than PDP-60. First, the overall performance of all competing systems are much lower than that of PDP-60. Second, incorporating supervised learning and expensive annotated knowledge bases to USSM provides insignificant gain this time (+3%), comparing to the large gain on PDP-60 (+19%).[7]

5.3 Customized training data for Winograd Schema Challenge

As previous systems collect relevant data from knowledge bases after observing questions during evaluation [25, 26], we also explore using this option. Namely, we build a customized text corpus based on questions in commonsense reasoning tasks. It is important to note that this does not include the answers and therefore does not provide supervision to our resolvers. In particular, we aggregate documents from the CommonCrawl dataset that has the most overlapping n-grams with the questions. The score for each document is a weighted sum of $F_1(n)$ scores when counting overlapping n-grams:

$$Similarity\_Score_{document} = \frac{\sum_{n=1}^{4} nF_1(n)}{\sum_{n=1}^{4} n}$$

[7] Our results so far have been with recurrent language models. As a comparison, we also trained a subword-level Transformer [37] LM on Wikipedia texts and obtain competitive performance (58.3% on PDP-60 and 54.1% on WSC-273).



One day when John and I had been out on some - business of our master 's , and were returning gently on a long , straight road , at some d- istance we saw a boy trying to leap a pony ove- r a gate ; the pony would not take the leap , - and the boy cut him with the whip , but he on- ly turned off on one side . He whipped him aga- in , but the pony turned off on the other side . Then the boy got off and gave him a hard t- hrashing , and knocked him about the head ...

Figure 2: **Left**: Histogram of similarity scores from top 0.1% documents in CommonCrawl corpus, comparing to questions in Winograd Schema Challenge. **Right**: An excerpt from the document whose score is 0.083 (highest ranking). In comparison, a perfect score is of 1.0. Documents in this corpus contain long series of events with complex references from several pronouns.

The top 0.1% of highest ranked documents is chosen as our new training corpus. Details of the ranking is shown in Figure 2. This procedure resulted in nearly 1,000,000 documents, with the highest ranking document having a score of 8×10^{-2}, still relatively small to a perfect score of 1.0. We name this dataset STORIES since most of the constituent documents take the form of a story with long chain of coherent events.

We train four different LMs on STORIES and add them to the previous ensemble of 10 LMs, resulting in a gain of 2% accuracy in the final system as shown in Table 5. Remarkably, single models trained on this corpus are already extremely strong, with a word-level LM achieving 62.6% accuracy, even better than the ensemble of 10 models previously trained on 4 other text corpora (61.5%).

Table 5: Accuracy on Winograd Schema Challenge, making use of STORIES corpus.

Method	Accuracy
USSM + Supervised DeepNet + Knowledge Base	52.8 %
Char-LM-*partial*	57.9%
Word-LM-*partial*	62.6%
Ensemble of 14 LMs	**63.7 %**

6 Analysis

6.1 Discovery of special words in Winograd Schema

We introduce a method to potentially detect keywords at which our proposed resolvers make decision between the two candidates $c_{correct}$ and $c_{incorrect}$. Namely, we look at the following ratio:

$$q_t = \frac{P_\theta(w_t|w_1, w_2, ..., w_{t-1}; w_k \leftarrow c_{correct})}{P_\theta(w_t|w_1, w_2, ..., w_{t-1}; w_k \leftarrow c_{incorrect})}$$

Where $1 \leq t \leq n$ for *full* scoring, and $k+1 \leq t \leq n$ for *partial* scoring. It follows that the choice between $c_{correct}$ or $c_{incorrect}$ is made by the value of $Q = \prod_t q_t$ being bigger than 1.0 or not. By looking at the value of each individual q_t, it is possible to retrieve words with the largest values of q_t and hence most responsible for the final value of Q.

We visualize the probability ratios q_t to have more insights into the decisions of our resolvers. Figure 3 displays a sample of incorrect decisions made by *full* scoring and is corrected by *partial* scoring. Interestingly, we found q_t with large values coincides with the special keyword of each Winograd Schema in several cases. Intuitively, this means the LMs assigned very low probability for the keyword after observing the wrong substitution. It follows that we can predict the keyword in each the Winograd Schema question by selecting top word positions with the highest value of q_t.

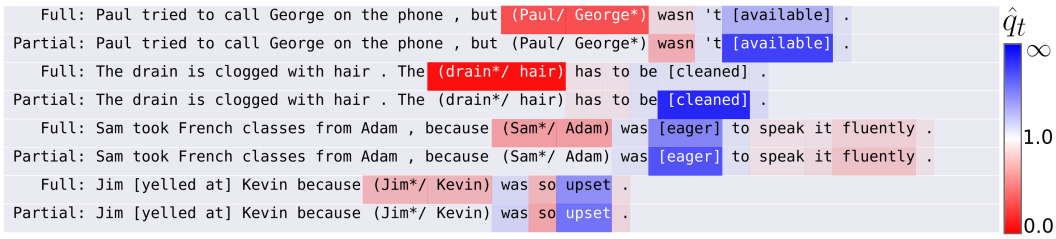


Figure 3: A sample of questions from WSC-273 predicted incorrectly by *full* scoring, but corrected by *partial* scoring. Here we mark the correct prediction by an asterisk and display the normalized probability ratio $\hat{q}_t$ by coloring its corresponding word. It can be seen that the wrong predictions are made mainly due to q_t at the pronoun position, where the LM has not observed the full sentence. *Partial* scoring shifts the attention to later words and places highest q values on the special keywords, marked by a squared bracket. These keywords characterizes the Winograd Schema Challenge, as they uniquely decide the correct answer. In the last question, since the special keyword appear before the pronoun, our resolver instead chose *"upset"*, as a reasonable switch word could be *"annoying"*.

Table 6: Accuracy of keyword detection from forward and backward scoring by retrieving top-2 words with the highest value of q_t

	Resolution accuracy	Special word retrieved
Forward scoring	63.7%	97 / 133
Backward scoring	58.2%	18 / 45

For questions with keyword appearing before the reference, we detect them by backward-scoring models. Namely, we ensemble 6 LMs, each trained on one text corpora with word order reversed. This ensemble also outperforms the previous best system on WSC-273 with a remarkable accuracy of 58.2%. Overall, we are able to discover a significant amount of special keywords (115 out of 178 correctly answered questions) as shown in Table 6. This strongly indicates a correct understanding of the context and a good grasp of commonsense knowledge in the resolver's decision process.

6.2 *Partial* scoring is better than *full* scoring.

In this set of experiments, we look at wrong predictions from a word-level LM. With *full* scoring strategy, we observe that q_t at the pronoun position is most responsible for a very large percentage of incorrect decisions as shown in Figfure 3 and Table 7. For example, with the test *"The trophy cannot fit in the suitcase because **it** is too big."*, the system might return $c_{incorrect}$ ="*suitcase*" simply because $c_{correct}$ = *"trophy"* is a very rare word in its training corpus and therefore, is assigned a very low probability, overpowering subsequent q_t values.

Table 7: Error analysis from a single model resolver. Across all three tests, *partial* scoring corrected a large portion of wrong predictions made by *full* scoring. In particular, it corrects more than 62.7% of wrong predictions on the Winograd Schema Challenge (WSC-273).

Data name	**Wrong prediction**	**Corrected**	**Correction percentage**
PDP-60	30	10	33.3%
PDP-122	55	33	60.0%
WSC-273	102	64	62.7%

Following this reasoning, we apply a simple fix to *full* scoring by normalizing its score with the unigram count of c: $Score_{full\ normalized} = Score_{full}/Count(c)$. *Partial* scoring, on the other hand, disregards c altogether. As shown in Figure 4, this normalization fixes *full* scoring in 9 out of 10 tested LMs on PDP-122. On WSC-273, the result is very decisive as *partial* scoring strongly outperforms the other two scoring in all cases. Since PDP-122 is a larger superset of PDP-60, we attribute the different behaviour observed on PDP-60 as an atypical case due to its very small size.

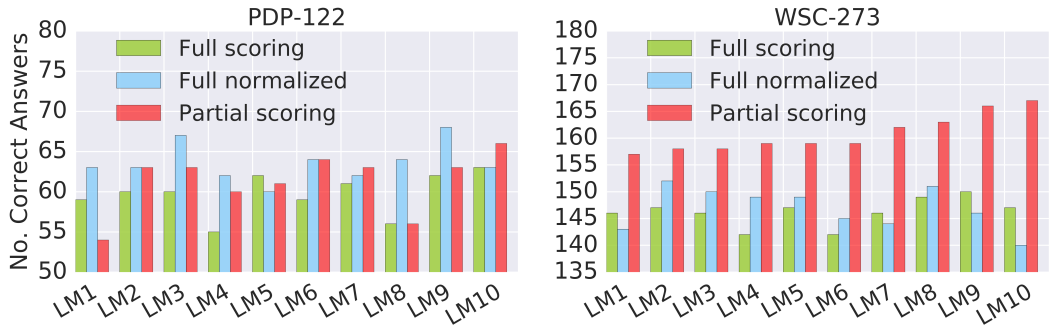


Figure 4: Number of correct answers from 10 different LMs in three modes *full*, *full normalized* and *partial* scoring. The second and third outperforms the first mode in almost all cases. The difference is most prominent on the largest test WSC-273, where *partial* scoring outperforms the other methods by a large margin for all tested LMs.

6.3 Importance of training corpus

In this set of experiments, we examine the effect of training data on commonsense reasoning test performance. Namely, we train both word-level and character-level LMs on each of the five corpora: LM-1-Billion, CommonCrawl, SQuAD, Gutenberg Books, and STORIES. A held-out dataset from each text corpus is used for early stopping on the corresponding training data.

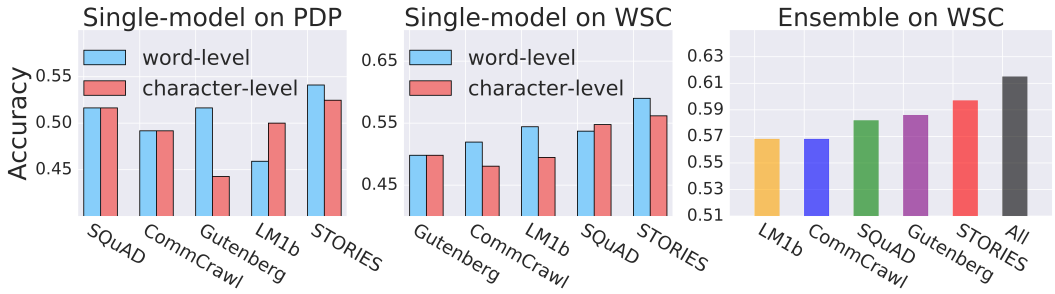


Figure 5: **Left and middle**: Accuracy of word-level LM and char-level LM on PDP-122 and WSC-273 test sets, when trained on different text corpora. **Right**: Accuracy of ensembles of 10 models when trained on five single text corpora and all of them. A low-to-high ranking of these text corpora is LM-1-Billion, CommonCrawl, SQuAD, Gutenberg, STORIES.

To speed up training on these large corpora, we first train the models on the LM-1-Billion text corpus. Each trained model is then divided into three groups of parameters: Embedding, Recurrent Cell, and Softmax. Each of the three is optionally transferred to train the same architectures on CommonCrawl, SQuAD and Gutenberg Books. The best transferring combination is chosen by cross-validation.

Figure 5-left and middle show that STORIES always yield the highest accuracy for both types of input processing. We next rank the text corpora based on ensemble performance for more reliable results. Namely, we compare the previous ensemble of 10 models against the same set of models trained on each single text corpus. This time, the original ensemble trained on a diverse set of text corpora outperforms all other single-corpus ensembles including STORIES. This highlights the important role of diversity in training data for commonsense reasoning accuracy of the final system.

7 Conclusion

We introduce a simple unsupervised method for Commonsense Reasoning tasks. Key to our proposal are large language models, trained on a number of massive and diverse text corpora. The resulting systems outperform previous best systems on both Pronoun Disambiguation Problems and Winograd Schema Challenge. Remarkably on the later benchmark, we are able to achieve 63.7% accuracy,

comparing to 52.8% accuracy of the previous state-of-the-art, who utilizes supervised learning and expensively annotated knowledge bases. We analyze our system's answers and observe that it discovers key features of the question that decides the correct answer, indicating good understanding of the context and commonsense knowledge. We also demonstrated that ensembles of models benefit the most when trained on a diverse set of text corpora.

We anticipate that this simple technique will be a strong building block for future systems that utilize reasoning ability on commonsense knowledge.

References

[1] Hector J Levesque, Ernest Davis, and Leora Morgenstern. The winograd schema challenge. In *AAAI spring symposium: Logical formalizations of commonsense reasoning*, 2011.

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012.

[3] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1701–1708, 2014.

[4] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *Advances in Neural Information Processing Systems*, 2015.

[5] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, et al. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[8] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

[9] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*, 2014.

[10] Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Mike Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig. Achieving human parity in conversational speech recognition. *arXiv preprint arXiv:1610.05256*, 2016.

[11] Chung-Cheng Chiu, Tara N Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J Weiss, Kanishka Rao, Katya Gonina, et al. State-of-the-art speech recognition with sequence-to-sequence models. *arXiv preprint arXiv:1712.01769*, 2017.

[12] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*, 2015.

[13] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112, 2014.

[14] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[15] Hany Hassan, Anthony Aue, Chang Chen, Vishal Chowdhary, Jonathan Clark, Christian Federmann, Xuedong Huang, Marcin Junczys-Dowmunt, William Lewis, Mu Li, et al. Achieving human parity on automatic chinese to english news translation. *arXiv preprint arXiv:1803.05567*, 2018.

[16] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[17] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[18] Andrew M Dai and Quoc V Le. Semi-supervised sequence learning. In *Advances in Neural Information Processing Systems*, pages 3079–3087, 2015.

[19] Prajit Ramachandran, Peter J Liu, and Quoc V Le. Unsupervised pretraining for sequence to sequence learning. In *Conference on Empirical Methods in Natural Language Processing*, 2017.

[20] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2018.

[21] Jeremy Howard and Sebastian Ruder. Fine-tuned language models for text classification. *arXiv preprint arXiv:1801.06146*, 2018.

[22] Haoruo Peng, Daniel Khashabi, and Dan Roth. Solving hard coreference problems. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 809–819, 2015.

[23] Dan Bailey, Amelia Harrison, Yuliya Lierler, Vladimir Lifschitz, and Julian Michael. The winograd schema challenge and reasoning about correlation. In *In Working Notes of the Symposium on Logical Formalizations of Commonsense Reasoning*, 2015.

[24] Peter Schüller. Tackling winograd schemas by formalizing relevance theory in knowledge graphs. In *Fourteenth International Conference on the Principles of Knowledge Representation and Reasoning*, 2014.

[25] Altaf Rahman and Vincent Ng. Resolving complex cases of definite pronouns: the winograd schema challenge. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 777–789. Association for Computational Linguistics, 2012.

[26] Arpit Sharma, Nguyen Ha Vo, Somak Aditya, and Chitta Baral. Towards addressing the winograd schema challenge-building and using a semantic parser and a knowledge hunting module. In *IJCAI*, pages 1319–1325, 2015.

[27] Quan Liu, Hui Jiang, Zhen-Hua Ling, Xiaodan Zhu, Si Wei, and Yu Hu. Combing context and commonsense knowledge through neural networks for solving winograd schema problems. *CoRR*, abs/1611.04146, 2016.

[28] Zewei Chu, Hai Wang, Kevin Gimpel, and David A. McAllester. Broad context language modeling as reading comprehension. *CoRR*, abs/1610.08431, 2016.

[29] Nasrin Mostafazadeh, Michael Roth, Annie Louis, Nathanael Chambers, and James Allen. Lsdsem 2017 shared task: The story cloze test. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*, pages 46–51, 2017.

[30] Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V Le. Qanet: Combining local convolution with global self-attention for reading comprehension. *arXiv preprint arXiv:1804.09541*, 2018.

[31] Rafal Józefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. Exploring the limits of language modeling. *CoRR*, abs/1602.02410, 2016.

[32] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[33] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

[34] Quan Liu, Hui Jiang, Zhen-Hua Ling, Si Wei, and Yu Hu. Probabilistic reasoning via deep learning: Neural association models. *CoRR*, abs/1603.07704, 2016.

[35] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[36] Hugo Liu and Push Singh. Conceptnet—a practical commonsense reasoning tool-kit. *BT technology journal*, 22(4):211–226, 2004.

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010, 2017.

[38] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

A Recurrent language models

The base model consists of two layers of Long-Short Term Memory (LSTM) [32] with 8192 hidden units. The output gate of each LSTM uses peepholes and a projection layer to reduce its output dimensionality to 1024. We perform drop-out on LSTM's outputs with probability 0.25.

For word inputs, we use an embedding lookup of 800000 words, each with dimension 1024. For character inputs, we use an embedding lookup of 256 characters, each with dimension 16. We concatenate all characters in each word into a tensor of shape *(word length, 16)* and add to its two ends the <*begin of word*> and <*end of word*> tokens. The resulting concatenation is zero-padded to produce a fixed size tensor of shape *(50, 16)*. This tensor is then processed by eight different 1-D convolution (Conv) kernels of different sizes and number of output channels, listed in Table 8, each followed by a ReLU acitvation. The output of all CNNs are then concatenated and processed by two other fully-connected layers with highway connection that persist the input dimensionality. The resulting tensor is projected down to a 1024-feature vector. For both word input and character input, we perform dropout on the tensors that go into LSTM layers with probability 0.25.

Table 8: One-dimensional convolutional layers used to process character inputs

	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5	Conv 6	Conv 7	Conv 8
Kernel size	1	2	3	4	5	6	7	7
Output channels	32	32	64	128	256	512	1024	2048

We use a single fully-connected layer followed by a $Softmax$ operator to process the LSTM's output and produce a distribution over word vocabulary of size 800K. During training, LM loss is evaluated using importance sampling with negative sample size of 8192. This loss is minimized using the AdaGrad [38] algorithm with a learning rate of 0.2. All gradients on LSTM parameters and Character Embedding parameters are clipped by their global norm at 1.0. To avoid storing large matrices in memory, we shard them into 32 equal-sized smaller pieces. In our experiments, we used 8 different variants of this base model as listed in Table 9.

Table 9: All variants of recurrent LMs used in our experiments.

LM name	Difference to base settings
Word-LM 1	Dropout rate 0.1
Word-LM 2	Learning rate 0.05
Word-LM 3	Residual connections around LSTM layers
Word-LM 4	Project dimension 2048, embedding dimension 2048, One layer of LSTM
Char-LM 1	Embedding dimension 4096, project dimension 2048
Char-LM 2	Embedding dimension 2048, project dimension 2048
Char-LM 3	Embedding dimension 1024, learning rate 0.1, Residual instead of Highway connection
Char-LM 4	Learning rate 0.002, Embedding dimension 1024

In Table 10, we listed all LMs and their training text corpora used in each of the experiments in Section 5.

Table 10: Details of LMs and their training corpus reported in our experiments.

Experiment	**LM variant / training corpus**
Single models on PDP-60	Word-LM 1/Gutenberg and Char-LM 1/Gutenberg
Ensemble on PDP-60	**Two single models on PDP-60 +** Word-LM 2/SQuAD + Char-LM 2/LM1B + Char-LM 3/CommonCrawl
Ensemble of 10 models on WSC-273	**Ensemble on PDP-60 +** Word-LM 1/Gutenberg *(different random seed)* + Word-LM 1/LM1B + Char-LM 4/Gutenberg + Char-LM 4/SQuAD + Char-LM 4/CommonCrawl
Ensemble of 14 models on WSC-273	**Ensemble of 10 models on WSC-273 +** Word-LM 1/STORIES + Char-LM 2/STORIES + Word-LM 3/STORIES + Word-LM 4/STORIES
Ensemble of 6 backward-scoring models on WSC-273	Word-LM 1/Gutenberg + Word-LM 1/STORIES + Char-LM 4/CommonCrawl + Char-LM 4/SQuAD + Word-LM 4/LM1B + Char-LM 2/STORIES +

B Data contamination in CommonCrawl

Using the similarity scoring technique in section 5.3, we observe a large amount of low quality training text on the lower end of the ranking. Namely, these are documents whose content are mostly unintelligible or unrecognized by our vocabulary. Training LMs for commonsense reasoning tasks on full CommonCrawl, therefore, might not be ideal. On the other hand, we detected and removed a portion of PDP-122 questions presented as an extremely high ranked document.