Towards Achieving Machine Comprehension Using Deep Learning on Non-GPU Machines

Uzair Khan
College of Computing and Information Sciences, PAF Karachi Institute of Economics and Technology,
Karachi, Pakistan
ukhan@eemaan.com

Khalid Khan
College of computing and Information Sciences, PAF Karachi Institute of Economics and Technology,
Karachi, Pakistan
khalid.khan@pafkiet.edu.pk

Fadzil Hasssan
Department of Computer and Information Sciences, Universiti Teknologi Petronas,
Perak, Malaysia
mfadzil_hassan@utp.edu.my

Anam Siddiqui
Department of Computer Sciences,
Sir Syed University of Engineering and Technology,
Karachi, Pakistan
sidd.anam@ssuet.edu.pk

Muhammad Afaq
College of Computing and Information Sciences, PAF Karachi Institute of Economics and Technology,
Karachi, Pakistan
sanaraba@gmail.com

Abstract—**Long efforts have been made to enable machines to understand human language. Nowadays such activities fall under the broad umbrella of machine comprehension. The results are optimistic due to the recent advancements in the field of machine learning. Deep learning promises to bring even better results but requires expensive and resource hungry hardware. In this paper, we demonstrate the use of deep learning in the context of machine comprehension by using non-GPU machines. Our results suggest that the good algorithm insight and detailed understanding of the dataset can help in getting meaningful results through deep learning even on non-GPU machines.**

Keywords-natural language processing; machine comprehension; deep learning; non-GPU machines; SQuAD

I. INTRODUCTION

Automated question answering has been a point of discussion in the field of computer science for more than 50 years and can easily be the most awaited task in the domain of natural language processing. Still it is considered as an open problem and research is still conducted on its various aspects. The early work in this area was domain specific in which AI was used as a building block [1]. With the advancement of enabling technologies, researchers found various ways/methods to come out of the domain restriction and focus on open domain systems and applications such as chat bots, personal assistants, medical assistants, etc. Today, automated question answering has become a part of the more holistic concept of Machine Comprehension which enables machines to analyze a context and answer general questions against it by utilizing new and evolving machine learning models. The goal is to achieve human-like accuracy on the said task. With the ever increasing processing capabilities and hardware capacity, advance machine learning techniques and deep learning algorithms, available knowledge bases and generalized data sets such as SQuAD [2, 3], it is expected that machine comprehension will soon achieve human like accuracy. Question answering can be broken down into three basic parts: understanding the question, searching for answer and answer generation. Understanding factoid questions is the simplest task and machines can usually carry one or two word answers, e.g. answering a question regarding the distance of Moon from Earth. For simple questions like this, we can use pattern matching techniques to find answers from the given text and the answer generation is also simple. Various open source libraries are available for this purpose [4]. Taming generally means to extract only the information which contains some meaning. This includes stemming, i.e. breaking down verbs in their first tense, and the ability to extract these factors with increasing amount of text.

II. CONVENTIONAL AND MODERN TECHNIQUES

Automated question answering has two common approaches, answer matching and machine learning. In answer matching, one needs to do information extraction. In a question like "how many white tigers are left in the world in 2018?", the keywords are white, tigers, left, world, 2018. The term "how many" determines that a quantity is asked, which is the asking-point of this question. As a first step, the question is tokenized by using a tokenization technique [5]. The second step is do token matching with corpus keywords that may result in the selection of a few sentences where the matching remains positive and there is a chance to find potential answers. The intersection of question and sentences gives a score which determines how likely it is to get the answer in any sentence. Machine learning on the other hand is a branch of artificial intelligence (AI) that gives computers the ability to automatically learn from the data and improve their efficiency without being explicitly programmed [6]. Mostly, machine learning models use supervised learning in which the system

Corresponding author: Khalid Khan

learns from given examples, so it has both input and output variables and use algorithms to derive mapping functions from input to output. So we can say that the data is labeled [7]. Whereas, in unsupervised learning, the algorithms themselves find relevant structures in the context, so there are no output variables. In unsupervised learning there are no correct answers and the machine has to draw inferences from datasets to describe hidden structures from data. Basically, the data is unlabeled. There is another approach which is known as semi-supervised learning falling right between supervised and unsupervised learning. It typically uses a small amount of labeled data and a large amount of unlabeled data to improve learning accuracy. Deep learning accomplishes results that were unrealistic earlier [8]. Deep learning models can accomplish intricate perfection sometimes even more accurate than human performance. Deep learning systems are instructed by utilizing large arrangements of labeled data and neural network architectures that learn features from the information without the requirement of manual feature extraction. A standout amongst the most prominent kinds of deep neural networks is known as convolutional neural networks (CNN or ConvNet). A CNN convolves learned features with fed data, and utilizes 2D convolutional layers, making this design appropriate to preparing 2D content, for example, pictures. There are three ways in order to categorize objects using deep learning:

A. Training from Scratch

To create a network architecture which learns the features we need to gather a very large dataset. This is a less common approach because it is good for the applications that have a large number of output categories or generally new applications. The problem is that with a large amount of data we typically need a few days or even weeks to train the model.

B. Transfer Learning

The transfer learning method is used by the majority of deep learning apps. This method includes the fine-tuning of an already trained model. One starts with an already existing network like AlexNet or GoogLeNet, and adds some new data in it which contain previously unknown classes. This also comes with an advantage of needing less data (operating thousands of pictures instead of millions), so the operating time reduces.

C. Feature Extraction

A less used, more concentrated way of deep learning is to utilize the network as a feature extractor. Since every one of the layers is instructed with learning specific features from pictures, we can extract these features from the network whenever amid the training procedure. Most of the researches on this task are performed on a cluster of computers. However we are trying to achieve it on mid-range computers, contrary to the fact that deep learning models require heavy computational power. To achieve this, we gave more time and smaller chunks of data as per the computational power we are providing to this task. We test results by partitioning the data differently until they provide us with the best possible results. This might be a new step towards deep learning with conventional computing power.

III. RELATED WORK

Automated question and answer using Wikipedia as a source has been discussed in [3]. Wikipedia is used as the knowledge base, bigram hashing and TF-IDF matching are performed with a multi-layer recurrent neural network model to detect answers in Wikipedia's paragraphs. Authors in [9] performed a similar task, matching potential answers on semantic similarity with the question and concluded that a bigram model performs better than the unigram model and the addition of the IDF-weighted word count features improves performance for both models by 10%–15%. Its efficiency can be further improved using a sentence model with higher order n-grams. Authors focused on answer sentence selection, the task that selects the correct sentence, answering a factual question from a set of candidate sentences. The relevance of an answer sentence to a question is typically determined by measuring the semantic similarity between question and answer, but in this paper authors shown that a neural network-based sentence model can be applied to the task of answer sentence selection that can be applied to any language and does not require feature-engineering and hand-coded resources beyond some large corpus on which to train the initial word embeddings [9].

The answer selection problem can be formulated as follows [10]: Given a question q and an answer candidate pool {a1,a2,•••,as} for this question, we aim to search for the best answer candidate by using bidirectional long short-term memory (biLSTM) models on both questions and answers respectively. In order to better distinguish candidate answers according to the question, authors in [10] introduced a simple but efficient attention model to this framework for the answer embedding generation according to the question context. In [11], grounded text and question-answer pairs are simultaneously generated with the hope that the analysis of performance on these tasks will help expose the frailty of current models and help motivate new algorithm designs that alleviate these frailties. Authors in [12] proposed a new end-to-end neural architecture to address the machine comprehension problem as defined in the SQuAD dataset [12]. They used the Pointer Net model that allows the predictions of tokens from the input sequence only, rather than from a larger fixed vocabulary and thus allows them to generate answers that consist of multiple tokens from the original text. In [13], authors built a reading comprehension dataset containing 500 fictional stories, with 4 multiple choice questions per story. The stories were chosen to be fictional to focus work on finding the answer in the story itself, rather than in knowledge repositories such as Wikipedia. The goal is to build technology that actually understands stories and paragraphs on a deep level such as opposed to using information retrieval methods and the redundancy of the web to find the answers.

Authors in [14] addressed the issue of unavailability of real natural language training data by introducing a novel approach for building a supervised reading comprehension data set. Supervised machine learning approaches have largely been absent from this space due to both the lack of large-scale training datasets, and the difficulty in structuring statistical models flexible enough to learn to exploit document structure.

Authors compared these neural models to a range of baselines and heuristic benchmarks based upon a traditional frame semantic analysis provided by a state-of-the-art natural language processing. In [15], authors introduced the Bi-Directional Attention Flow (BIDAF) network, a hierarchical multi-stage architecture for modeling the representations of the context paragraph at different levels of granularity. The computed attention weights were used to extract the most relevant information from the context for answering the question by summarizing the context into a fixed-size vector. This allowed the attention at each time step to be unaffected from previous incorrect attendances [15]. In [3], the authors addressed the need for a large and high-quality reading comprehension dataset by presenting Stanford Question Answering Dataset v1.0 (SQuAD) [2], consisting of questions posed by crowd workers on a set of Wikipedia articles, where the answer to every question is a segment of text, or span, from the corresponding reading passage. The authors also found that the model performance worsens with increasing complexity of (i) answer types and (ii) syntactic divergence between the question and the sentence containing the answer. Another solution to the problem on machine comprehension is long short-term memory (LSTM) architecture for Recurrent Neural Networks (RNN) which is used to find complex answers using semantics [16]. Machine reading, text extraction and word embedding is used to improve the understanding the text along with LSTM and RNN [17]. Almost all the discussed models were trained using GPU machines to ensure accuracy, although machine learning models can work on a moderate non-GPU machine as well. In this paper, we have shown that a deep learning model using the same dataset of Wikipedia can be trained over ordinary CPU based machines, if the data-set does not contain images or large data entities. SQuAD [2] is comparatively a mid-size data-set containing 100,000+ question-answer pairs on 500+ articles. We used a 3rd generation Intel Powered machine with 8Gb ram for this testing (no graphics memory).

IV. Methodology

We used deep learning methods on a non-GPU machine to create a model that predicts answers from a given context. As mentioned previously, SQUAD is a new generation dataset which contains questions by a number of people on Wikipedia articles. The answer of each question is always a part of a given context and about three-quarters of the answers are four words long. There is a fast progress in this dataset where some models achieve the same accuracy as humans in the task of question answering.

A. Embedding

GloVe (Global Vectors) [18] embedding is an unsupervised learning algorithm to obtain vector representation of words and the resulting representations showcase interesting linear substructures of the word vector space. For this problem there were used 100 GloVe word embeddings.

B. Encoding

We used bi-directional LSTM to add a RNN based encoding layer, because we wanted each word to be aware of words before and after it. The output is a series of vectors in the forward and backward direction which were we concatenated.

C. Attention Layer

Since we got the hidden context vector and the hidden question vector, for answer generation we have to look at them together. That's where the attention layer comes in. It is the primary component of the Question Answering Task.

D. Dot Production Attention

For each context vector c_i we multiply each question vector q_j to get vector e_i (attention scores). Then we take a softmax over e_i to get α_i (attention distribution). Softmax ensures that the sum of all e_i is 1. Finally we calculate α_i as the product of the attention distribution α_i and the corresponding question vector. Dot product attention is described below:

$$e^i = [c_i^T q_1, \dots, c_i^T q_M] \epsilon R^M \quad (1)$$

$$\alpha^i = softmax(e^i)\ \epsilon R^M \quad (2)$$

$$a^i = \sum_{j=1}^{M} \alpha_j^i q^j \in R^{2h} \quad (3)$$

We run SQuAD with basic attention layer as mentioned, but results were not as promising. For better results we needed to add more complexity. As mentioned above, Bi-Directional Attention Flow can improve the results i.e. computing the similarity matrix, which contains a similarity score S_{ij} for each pair (c_i, q_j) of context and question hidden states, as shown in (4):

$$S_{ij} = w_{sim}^T \left[c_i; q_j; c_i \circ q_j\right] \epsilon R \quad (4)$$

where, $c_i \circ q_j$ is an element-wise product and w_{sim} is a weight vector. Similar to the dot product attention above, we performed Context to Question Attention using softmax of S to get Attention α_i , which is used later to get C2Q attention outputs α_i, from the weighted sums of question hidden states q_j.

$$\alpha^i = softmax\left(S_{i,z}\right) \epsilon R^M \ \ \forall i \epsilon \{1, \dots, N\} \quad (5)$$

$$a_i = \sum_{j=1}^{M} \alpha_j^i q_j \epsilon R^{2h} \ \ \forall i \epsilon \{1, \dots, N\} \quad (6)$$

For Question to Context Attention, for each context location $\{1,\dots,N\}$, we take the max of the corresponding row of similarity matrix, $m_i = \max j\ S_{ij} \in R$. Then the softmax over the resulting vector $m \in R^N$ gives the attention distribution $\beta \in R^N$ of context locations. Then, β is used to get the weighted sum of context hidden states c_i. This is the Question to Context attention prime output.

$$m_i = S_{ij} \ \in R \ \ \forall i \epsilon \{1, \dots, N\} \quad (7)$$

$$\beta = softmax(m)\ \epsilon R^N \quad (8)$$

$$c' = \sum_{i=1}^{N} \beta_i c_i\ \epsilon R^{2h} \quad (9)$$

Eventually, to get context position c_i, we combine both outputs i.e. Context to Question and Question to Context as:

$$b_i = [c_i; a_i; c_i \circ a_i; c_i \circ c']\ \epsilon R^{8h}\ \forall i \epsilon \{1, \dots, N\} \quad (10)$$

E. Output Layer

Using the results of the context hidden states and the attention vector from the previous layer we can decide the start and end index of the answer. So we created blended reps, these reps become the input to a fully connected soft-max layer to create a start vector with a probability of start index, and an end vector with a probability of end index. Loss function is calculated using cross-entropy loss for start index, summed with cross-entropy loss for end index. The results were evaluated using F1 score and EM score. F1 score is the weighted average of Precision and Recall. Precision is the ratio of correctly predicted positive answers to total predicted positive observations, whereas Recall is the ratio of correctly predicted positive answers to total observations. EM score is the exact match score, the total number of predictions that match the actual answer.

V. RESULTS

The test results were quite appealing, but we could not run more than 3 epochs due to exhaustion of resources. It took more than 24 hours to complete 1000 iterations or 1 epoch which provided us with the following results. Our average F1 score turned out to be 0.77, with average EM score of 0.64. The average loss was 3.2. The results of 12 iterations are shown in Figures 1 and 2. Figure 1 shows the outcome of 12 iterations, Blue color shows the F1 score and orange shows the EM score, Grap range is from 0 to1, 1 being the best case. Figure 2 shows the loss in each of the 12 iterations. Loss is inversely proportional to the correctness of the model. The more the loss, the less accurate is the model. Observed values were around 3.2 which means that the model performs accurately.

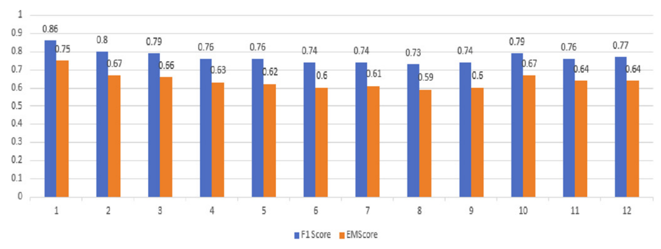

Fig. 1. F1 and EM score

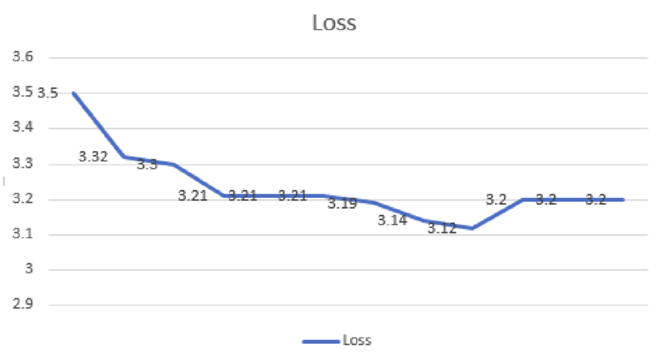

Fig. 2. Loss

VI. CONCLUSION

Machine learning research aims to produce methods that can reason with natural language. Deep learning is playing an important role in this field improving the results. We have demonstrated the use of deep learning in the context of machine comprehension by using non-GPU machines. Our results suggest that the good insight of the algorithm and the detailed understanding of the dataset can help in getting meaningful results through deep learning even on non-GPU machines. We believe existing learning systems cannot solve many issues in this field, whereas our proposed method has produced results with reasonable improvements.

Our results show that, the performance of the model is quite appealing, however for a moderate sized dataset like SQuAD, these models need more processing power. The results are not bad in one epoch, but ultimately, to train these data sets, more powerful computers are needed to train deep learning models. There might be a follow up in the future in this direction which may provide us with even better results.

REFERENCES

[1] D. Karunakaran, "Entity extraction using Deep Learning based on Guillaume Genthial work on NER", available at: https://medium.com/intro-to-artificial-intelligence/entity-extraction-using-deep-learning-8014acac6bb8, 2017

[2] https://rajpurkar.github.io/SQuAD-explorer/

[3] P. Rajpurkar, J. Zhang, K. Lopyrev, P. Liang, "SQuAD: 100,000+ Questions for Machine Comprehension of Text", available at: https://arxiv.org/abs/1606.05250, 2016

[4] D. Chen, A. Fisch, J. Weston, A. Bordes, "Reading Wikipedia to Answer Open-Domain Questions", 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, Canada, July 30-August 4, 2017

[5] E. Loper, S. Bird, "NLTK: The natural language toolkit", ACL-02 Workshop on Effective tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics, Philadelphia, USA, July 7, 2002

[6] M. Richardson, C. J. C. Burges, E. Renshaw, "MCTest: A Challenge Dataset for the Open-Domain Machine Comprehension of Text", Conference on Empirical Methods in Natural Language Processing, Washington, USA, October 18-21, 2013

[7] J. Weston, A. Bordes, S. Chopra, A. M. Rush, B. V. Merrienboer, A. Joulin, T. Mikolov, "Towards AI-Complete Question Answering: A Set of Prerequisite Toy Tasks", 4rth International Conference on Learning Representations, New York, USA, May 2-4, 2016

[8] K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, P. Blunsom, "Teaching Machines to Read and Comprehend", International Conference on Neural Information Processing Systems, Montreal, Canada, December 7-12, 2015

[9] S. B. Kotsiantis, "Supervised machine learning: A review of classification techniques", Informatica, Vol. 31, pp. 249-268, 2007

[10] J. Schmidhuber, "Deep learning in neural networks: An overview", Neural Networks, Vol. 61, pp. 85-117, 2015

[11] B. F. Green Jr, A. K. Wolf, C. Chomsky, K. Laughery, "Baseball: An Automatic Question-Answerer", Western Joint IRE-AIEE-ACM Computer Conference , Los Angeles, California, May 9-11, 1961

[12] M. Seo, A. Kembhavi, A. Farhadi, H. Hajishirzi, "Bidirectional Attention Flow for Machine Comprehension", 5th International Conference on Learning Representations, Toulon, France, April 24-26, 2017

[13] Y. LeCun, Y. Bengio, G. Hinton, "Deep Learning", Nature, Vol. 521, Article ID 7553, 2015

[14] K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink, J. Schmidhuber, "LSTM: A search space odyssey", Transactions on Neural Networks and Learning Systems, Vol. 28, No. 10, pp. 2222-2232, 2017

[15] L. Yu, K. M. Hermann, P. Blundom, S. Pulman, "Deep learning for answer sentence selection", available at: https://arxiv.org/pdf/1412.1632.pdf, 2014

[16] A. Finch, Y. S. Hwang, E. Sumita, "Using machine translation evaluation techniques to determine sentence-level semantic equivalence", Third International Workshop on Paraphrasing, Jeju Island, Korea October 14, 2005

[17] K. Cho, B. V. Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, "Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation", Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, October 25-29, 2014

[18] J, Pennington, R. Socher, C. D. Manning, "GloVe: Global Vectors for Word Representation", Conference on Empirical Methods in Natural Language Processing Doha, Qatar, October 25-29, 2014