Improving Question Answering with an Ensemble Approach

Cong Bao
University College London
ucabcba@ucl.ac.uk

Yuan Gao
University College London
ucabyg5@ucl.ac.uk

Yuan Li
University College London
ucabiox@ucl.ac.uk

Abstract

Question answering is a challenge task in Natural Language Understanding (NLU). Prior works suggesting the feasibility of combining different feature representations to obtain more abundant information from input documents. In our work, we leverage both context-dependent and context-independent representations learned from pre-trained models to build a robust ensemble model. We experiment on CommonsenseQA dataset and the evaluation results showing the ensemble approach outperforms individual feature representation methods, indicating the feasibility of ensemble approach in question answering tasks.

1 Introduction

Question Answering (QA) is a major challenge in Natural Language Understanding (NLU) that an automatic system is expected to be designed to response questions asked in natural language using either a pre-structured database or a collection of natural language documents (Chali et al., 2011; Dwivedi and Singh, 2013; Ansari et al., 2016; Lende and Raghuwanshi, 2016). Prior works that attempted using both contextual-independent representations (Kumar et al., 2015; Suggu et al., 2016) and contextual-dependent representations (Peters et al., 2018; Vaswani et al., 2017) as features of input sentences achieve desirable outcomes in various tasks or datasets (Ostermann et al., 2018; Rajpurkar et al., 2016a, 2018). Ensemble approaches (Ma and Hovy, 2016; Liu et al., 2017), among prior works, outperform other models with single feature extraction methods significantly, indicating the importance of combing different representations of features to enhance performance.

Inspired by this, in our work, we combine both context-dependent representations and context-independent representations of sentences as features to downstream tasks. To achieve this, we transfer pre-trained feature based model such as ELMo (Peters et al., 2018) and fine-tuning based model such as BERT (Vaswani et al., 2017) to extract context-dependent representations of inputs, learning contextual features from concatenated question and answer pairs. In addition, we leverage pre-trained word embeddings such as GloVe (Pennington et al., 2014) as context-independent representations of sequences. We separately pre-training or fine-tuning these models on target task, and concatenate the learned feature vectors to fine-tuning the whole ensemble model on the task.

The target dataset we used is CommonsenseQA (Talmor et al., 2018), which contains 9,500 multiple choice questions with five given alternatives including only one actual answer. There are two types of data split, which are "random split" and "question token split". The main evaluation split is Random split, where each set of data contains questions with uniformly randomly split concepts. In question token split, each set have disjoint question concepts. In our work, random split data is mainly used in fine-tuning.

The evaluation results showing the ensemble model outperforms single feature representation models in development datasets. Additionally, when removing one of components in the proposed ensemble model, the performance decreases in different degrees. Moreover, large scale pre-trained model such as BERT extract more essential information from raw texts than word embeddings, but still can be improved by weaker feature extractor, indicating the complementarity between different methods pre-training on different datasets and the importance of ensemble approaches.

2 Related Work

Ensemble learning is a machine learning paradigm which enables multiple learners trained to solve the same problem. This approach helps to integrate strength of different learners and generally has better performance than any single combined learners (Zhou, 2015; Surdeanu and Manning, 2010). Inspired by ensemble learning, within Natural Language Processing (NLP) domain, a considerable number of prior proposed works attempted to combine variance NLP models together and leveraged these ensemble models in several specific NLP problems.

Wang et al. (2018) presented a novel platform named General Language Understanding Evaluation (GLUE) for the purpose of evaluate the performance of models in natural language understanding(NLU) tasks. In their work, they ensembled BiLSTM (Hochreiter and Schmidhuber, 1997) with variance combinations of ELMo (Peters et al., 2018), CoVe (McCann et al., 2017) as well as Attention (Vaswani et al., 2017) models as baseline. They trained these ensemble models on different datasets with both single-task training and multi-task training strategies and then evaluated these models' performance by the platform they proposed. According to their experiment results, whatever the strategies, in most of datasets the performance of ensemble models is better than single BiLSTM model (Wang et al., 2018). Relate to our work, Wang et al. (2018) gives out the initial idea of our model, they ensembled BiLSTM with several context-dependent models and evaluated their performance. The idea of model ensemble they presented is valuable to refer to. Differently, taking account with our work, beside the context-dependent model, we leverage context-independent model within our ensemble model as well, which means that our model is able to extract and keep extra information from pre-trained data, this is what Wang et al. (2018) does no presented.

Similar to our work, Peters et al. (2018) tested pre-trained ELMo model they proposed on SQuAD (Rajpurkar et al., 2016b) dataset. This dataset forms of question and answering style and contains over 100K crowd sourced question and answer pairs where the answer is a span in a given Wikipedia paragraph. They leveraged model from Clark and Gardner (2017) as baseline model which is an improvement of Bidirectional Attention Flow model proposed by Seo et al. (2016). As described by the experiment results of Peters et al. (2018), single baseline model on the test set can reached 81% accuracy. While combing ELMo with the baseline model, the accuracy on the test set is 85.8% which increased by 4.8%. In their work, they merge their ELMo model with the original baseline model, which is trained based on GloVe (Pennington et al., 2014). Similar to our work, the combined model created by them can be considered as a hybrid model which contains both context-dependent and context-independent representations. Nevertheless, considering ELMo model is a feature-based model, which uses tasks-specific architectures that include the pre-trained representations as additional features, it means that its pre-trained parameters are frozen, only limited number of scalars can be fine-tune to fit down-stream task. Thus, it has potential limitation on fitness. Whereas, the model we proposed applied a fine-tuning based model named BERT (Devlin et al., 2018). Since BERT is a fine-tuning based model, it supports fine-tuning process on its pre-trained parameters. Empirically, comparing with single ELMo model, our ensemble model achieves better performance on downstream tasks.

3 Background

3.1 Question Answering Models

Question Answering systems in information retrieval are tasks that can automatically response to questions asked in natural language using either a pre-structured database or a collection of natural language documents (Chali et al., 2011; Dwivedi and Singh, 2013; Ansari et al., 2016; Lende and Raghuwanshi, 2016). According to this property, typically used question answering systems are Natural Language Processing QA and Hybrid QA.

3.2 Context-Dependent and Context-Independent Models

Context dependent models are models adjust their parameters based on data feed in to it, such as ELMo (Peters et al., 2018). It adjust its pre-trained parameters by data it accepted and learning semantics of words rely on this. Context independent models are models have freeze pre-trained parameters, their parameters do not change along with its training process. GloVe is an typical example of context independent model. It keeps its pre-trained parameters in constant and will not

change in downstream tasks (Peters et al., 2018).

3.3 Pre-Trained Language Representations

In this section, we discuss some pre-trained language representations and the approaches to apply them to downstream tasks in NLP, which are foundations of our model. There are generally two strategies of such approaches: feature-based and fine-tuning (Devlin et al., 2018).

3.3.1 Feature-based Strategy

GloVe (Pennington et al., 2014) is an pre-trained approach aims to obtaining vector representations for words based on unsupervised learning. It is trained by non-zero entries of a global word-word co-occurrence matrix, which tabulates how frequently words co-occur with one another in a given corpus. Benefit from this, GloVe can extract and keep information from common sense. By applying this speciality in our work, the model we designed can get extra information apart from given data set. ELMo (Peters et al., 2018) is another approach that extracts context-dependent features from a bidirectional language model. It is designed to enhance performance of existing models by extracting contextual embeddings from pre-trained model as additional features, or fine-tuning in specific downstream tasks.

3.3.2 Fine-tuning Strategy

A modern way to represent natural language is transferring models pre-trained on large dataset to specific tasks, with little adaptation in architecture and few parameter to learn. Researches in computer vision indicate the importance of transfer learning, where a typical example is fine-tuning models pre-trained on ImageNet (Deng et al., 2009; Yosinski et al., 2014). In NLP subject, an empirically powerful model based on fine-tuning is BERT (Devlin et al., 2018), which is pre-trained on BooksCorpus (800M words) and English Wikipedia (2,500M words). It uses a bidirectional Transformer to learn representations jointly conditioned on both left and right context in all layers. Our work is mainly based on BERT to extract contextual features from question and answer pairs. Therefore, we discuss BERT in some details to specify properties we are taking advantages of.

During the pre-training procedure of BERT, a next sentence prediction task is performed to learn relationships between two text sentences. In this task, two sentences `A` and `B` are concatenated as an single input, with two `[SEP]` marks at the end of each sentence and a `[CLS]` mark inserted before the first token in input. During training, there is a probability of 0.5 that `B` is the actual sentence that follows `A`. In other cases, `B` is a random sentence from the corpus. The `[CLS]` token is used as an output for classification. This setting in pre-training is of importance to our commonsense QA task. In our task, the answers are presented as multiple choices, which is similar to the next sentence prediction task that `A` is the question and `B` is a sample from answer set. Hence, we can easily transfer BERT to our task and fine-tune learned parameters in pre-trained model to extract sentence relationship information.

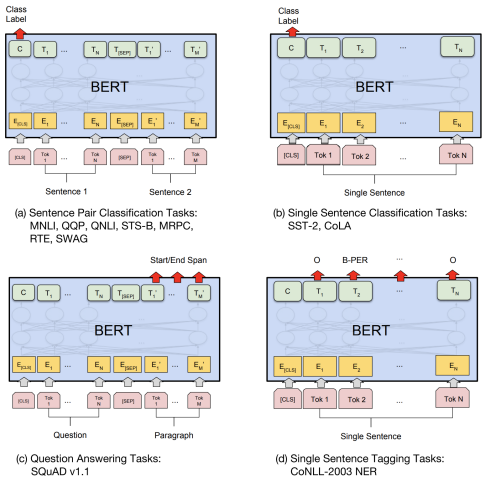

Figure 1: BERT Fine-Tuning for Down-Stream Tasks(Devlin et al., 2018)

BERT Transformer The basic component of BERT is Transformer encoder (Vaswani et al., 2017), in which self-attention is used to represent a sequence by relating different positions in the single sequence. Transformer encoders are listed in layers within BERT model, In our work since we are using basic version of BERT, thus, the number of layers $L = 12$, and hidden size $H = 768$, total parameters we have is $110M$

BERT Fine-tuning Figure 1 shows the fine-tune method of BERT for variance down-stream tasks. Within it, E represents the input embedding, T_i represents the contextual representation of token i, $[CLS]$ is the special symbol for classification output, and $[SEP]$ is the special symbol to separate non-consecutive token sequences.

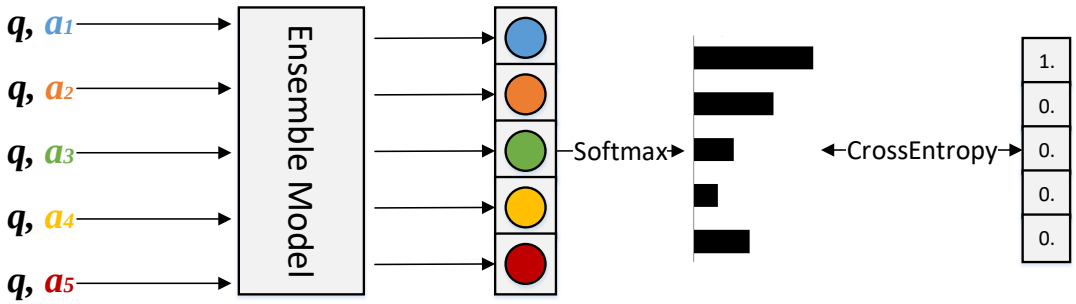

Figure 2: The structure of question answering multiple choice model. Five question ($\boldsymbol{q}$) and answer ($\boldsymbol{a}_i$) pairs are fed into a shared ensemble model. The result logit units are concatenated to yield the softmax distribution, which is used to obtain the loss by calculating cross-entropy between predictions and given labels.

4 Methods

Multiple choice question answering can be treated as a multi-class classification task. Assume there is a set of questions $\mathcal{Q}$, where each question $\boldsymbol{q}_k \in \mathcal{Q}$ is associated with a set of alternative answers $\{\boldsymbol{a}_{k1}, \boldsymbol{a}_{k2}, \ldots, \boldsymbol{a}_{kn}\}$, along with the judgements $\{y_{k1}, y_{k2}, \ldots, y_{kn}\}$. If answer $\boldsymbol{a}_{ki}$ is the correct response to question $\boldsymbol{q}_k$, the label $y_{ki} = 1$, otherwise $y_{ki} = 0$. Some prior works (Yu et al., 2014; Tan et al., 2015) group the question-answer-label pairs as a triple $(\boldsymbol{q_k}, \boldsymbol{a}_{ki}, y_{ki})$, and treat the task as a binary classification problem over each triple. This approach has a potential problem that the learned classifier can achieve an accuracy of 80% if it always returns false in a task with only one correct answer over five alternative responses. To avoid this problem, we develop our multiple choice model based on the work of Radford (2018), which is shown in Figure 2. We group the question-answer pairs as a list of triples $[(\boldsymbol{q}_k, \boldsymbol{a}_{k1}), (\boldsymbol{q}_k, \boldsymbol{a}_{k2}), \ldots, (\boldsymbol{q}_k, \boldsymbol{a}_{kn})]$ for each question $\boldsymbol{q}_k$ and its alternative answers $\boldsymbol{a}_{ki}$, and feed them into a shared model, which yields n logit units $[u_{k1}, u_{k2}, \ldots, u_{kn}]$, where $u_{ki} \in \mathbb{R}^1$. The resulting logit units are then concatenated as a single vector $\boldsymbol{u}_k \in \mathbb{R}^n$ with a softmax function performed on them to generate a valid prediction distribution. The loss is calculated as cross-entropy between prediction distribution and label distribution, and gradients are back-propagated through the shared model to update weights.

The shared model is an ensemble of three pre-trained models or embeddings, which are **AV**era**G**e pooling over bag-of-words (AVG), **E**mbeddings from **L**anguage **Mo**dels (ELMo; Peters et al., 2018), and **B**idirectional **E**ncoder **R**epresentations from **T**ransformers (BERT; Devlin et al., 2018). We denote the feature vector of three models as $\boldsymbol{f}_{AVG}$, $\boldsymbol{f}_{ELMo}$, and $\boldsymbol{f}_{BERT}$ respectively. The proposed ensemble approach is aiming to combine both context-independent representations and context-dependent representations as a more robust representation of input sentences for classification task. Among the three models, AVG provides a simple way to generate context-independent representation of a sentence by averaging pre-trained word embeddings, such as GloVe (Pennington et al., 2014), of all tokens in the sentence. Differently, BERT is a powerful fine-tuning based model to learn context-dependent representations. It is based on Transformer (Vaswani et al., 2017) encoder and trained on BooksCorpus (800M words; Zhu et al., 2015) and Wikipedia (2,500M words). In addition, ELMo learns a set of bidirectional Long Short Term Memory (LSTM; Hochreiter and Schmidhuber, 1997) over the 1B Word Benchmark (Chelba et al., 2013) to represent contextual information of a sentence. Although the learned hidden representations in ELMo are frozen, we can still fine-tune on the scalars to obtain a better representation on our specific task.

Details of extracting features from the three pre-trained models are described in the following subsections.

4.1 Average Pooling over Bag-of-words (AVG)

Given a token to embedding mapping function $\boldsymbol{S}$, as shown in (1), the sentence representation $\boldsymbol{f}_{AVG}$ can be calculated by summing over the embeddings of all tokens t in a sentence s, and then normalized by the valid length of sentence $|s|$ (num-

ber of tokens except padding symbols).

$$\boldsymbol{f}_{AVG} = \frac{1}{|s|} \sum_{t \in s} \boldsymbol{S}(t) \quad (1)$$

This approach does not consider the order of tokens, therefore it is a bag-of-words model. Here we only take into account a general context-independent representation of sentence, and do not include context-dependent information such as constitution and co-reference.

4.2 Embeddings from Language Models (ELMo)

The pre-trained ELMo can be used to fit on sentences in our task. As shown in (2), the representation of a sentence is calculated by summing all hidden layers $\boldsymbol{h}_i^{ELMo}$ in ELMo, scaling by trainable factors γ_i to re-weight the internal representations. In addition, we wrap a non-linear transformation over the sentence representation with a weight matrix $\mathbf{W}$, a bias vector $\boldsymbol{b}$, and an activation function `tanh`.

$$\boldsymbol{f}_{ELMo} = tanh\left(\mathbf{W} \cdot \sum_{i=1}^{L} \gamma_i \boldsymbol{h}_i^{ELMo} + \boldsymbol{b}\right) \quad (2)$$

ELMo helps to extract context-dependent features from a bidirectional language model. We can learn the parameters $\mathbf{W}$ and $\boldsymbol{b}$ as well as the factors γ on our task to enhance the sentence contextual representation. As the hidden representations $\boldsymbol{h}^{ELMo}$ is frozen, we will not modify the learned internal layers.

4.3 Bidirectional Encoder Representations from Transformers (BERT)

n BERT, there is a special classification token `[CLS]` in the first position of every sentences. As shown in (3), we take the hidden vector of the special token as the representation of input sentence, which is also wrapped with a non-linear transformation similar to ELMo.

$$\boldsymbol{f}_{BERT} = tanh\left(\mathbf{W} \cdot \boldsymbol{h}^{[\texttt{CLS}]} + \boldsymbol{b}\right) \quad (3)$$

Since the non-linear transformation is pre-trained as well, we can fine-tune the whole model in our task to generate more specific context-dependent representation of the input sentences.

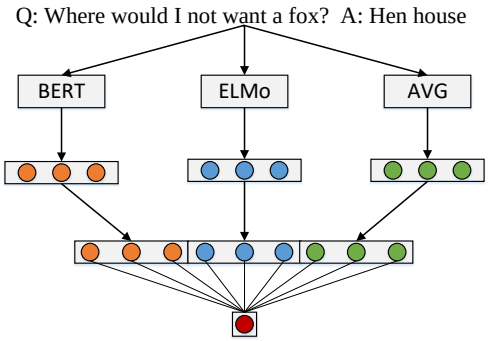

Figure 3: Structure of ensemble model.

4.4 Ensemble Approach

After fine-tuning on the three pre-trained models, as shown in (4), we concatenate the feature vectors of each model as an ensemble representation of a sentence.

$$\boldsymbol{f}_{COMB} = \mathbf{W} \cdot [\boldsymbol{f}_{AVG}; \boldsymbol{f}_{ELMo}; \boldsymbol{f}_{BERT}]^\top + \boldsymbol{b} \quad (4)$$

Similar to ELMo and BERT, we add a non-linear transformation to the sentence representation. However, the difference is that the weight matrix $\mathbf{W}$ has a dimensionality of $\mathbb{R}^{D \times 1}$, where D is the dimension of concatenated feature vector, and the bias vector $\boldsymbol{b} \in \mathbb{R}^1$. The model is intuitive in Figure 3. The 1-dimension vector output is treated as the logit unit, which will be concatenated with logit units of other sentences under a same question to calculate the softmaxed prediction distribution.

5 Experiments

The original model of ELMo and BERT are built based on Tensorflow Hub implementations. In addition, we use pre-trained GloVe as the word embeddings used in AVG model. The transformation matrix applied in ELMo has a output dimension of 256, which is a suitable size we found in experiments. For BERT, we keep the original hidden size 768 as the transformation has been pre-trained along with other parameters in the model. Apart from the above mentioned settings, other training or evaluation techniques used in our experiments are included in following subsections. Details of hyperparameters of best-performed models are introduced in Table 1.

Optimizer: Adam (Kingma and Ba, 2014) is

Model \ Param	hidden size	dr	lr	bs	epochs	parameters	accuracy(%)
AVG	300	0.05	10^{-5}	64	1	3,879,301	17.88
ELMo	256	0.05	10^{-3}	64	9	525,317	33.82
BERT	768	0.1	10^{-5}	32	3	110,105,659	54.37
AVG+ELMo	1068	0.05	10^{-3}	64	9	4,667,273	34.06
AVG+BERT	1068	0.05	10^{-5}	32	3	113,984,959	55.58
ELMo+BERT	1024	0.05	10^{-5}	32	3	110,630,975	56.32
ALL	1324	0.05	10^{-5}	32	3	114,247,619	57.36

Table 1: Hyperparameters and accuracy of best performed models. hidden size=dimension of feature layer; dr=dropout rate; lr=learning rate; bs=batch size.

used as our optimization approach with default parameter settings.

Learning rate: We set an initial learning rate of 1×10^{-5} when training BERT and ensemble in order to avoid over-fitting, as these modes have large amounts of parameters (over 110 million). For ELMo and AVG, we set a learning rate of 1×10^{-3}.

Learning rate decay: Apart from the learning rate adaption of Adam, we use an additional step learning rate decaying during training. It linearly drops the learning rate every step, which is useful in fine-tuning phase.

Gradient clipping: We use clip the gradients when the L_2 norm of gradients exceeds 1. It can limit the magnitude of the gradient and can make stochastic gradient descent (SGD) behave better in the vicinity of steep cliffs.

Regularization: L_2 regularization is appended to loss function of all the weights in output layers (layers on top of original models) with $\lambda = 10^{-3}$.

Warm up: Before the experiments started, a warm up session was performed to exclude first few steps which returns noisy loss and accuracy records. Warm up rate is set to 0.05.

Dropout: A small amount of dropout (5% to 10%) is applied to output layers to avoid over-fitting.

.

6 Evaluations

6.1 CommonsenseQA Dataset

CommonsenseQA is a multiple-choice question answering dataset that requires different types of commonsense knowledge to predict the correct answers. It contains $12,247$ questions with one correct answer and four distract answers. The dataset is provided in two major training/validation/testing set splits: "Random split" which is the main evaluation split, and "Question token split" where each of the five sets have disjoint question concepts. According to the original evaluation step, we split the data into a training/validation/test set with an $80/10/10$ split randomly(Talmor et al., 2018).

6.2 Baselines

Our ensemble is basing on different kinds of single model. To evaluate ensemble model, we consider single model as our baseline.(i.e. AVG, ELMo, BERT)

Bag-of-words Average Pooling: Average Pooling over Bag-of-words(AVG) is using the normalized the length of all tokens as a feature(Pennington et al., 2014). However doesn't consider the relationship of tokens.

ELMo: **E**mbeddings from **L**anguage **M**odels (ELMo) is a deep contextualized word representation(Peters et al., 2018). They can be easily added to existing models and enhance the sentence contextual representations.

BERT: BERT, which stands for **B**idirectional **E**ncoder **R**epresentaions from **T**ransformers, is consist of multi-layer bidirectional Transformer encoder based on the original im-

plementation described in (Vaswani et al., 2017). In order to obtain a fixed-dimensional pooled representation of the input sequence, we will fine-tune BERT to create contextualized word embeddings in ensemble model.

6.3 Results

Table 1 summarizes the performance as well as hyperparameter settings of both single and ensemble models. For each single model, we evaluate the accuracy on development set every 77 steps (granulity in our source code), and train it for 15 epochs until it is converged. For ensemble model, we stop the training early as the amount of training data is smaller than the overall trainable parameters, which means it is easy for the model to over-fitting.

Figure 4 gives an intuitive results of the evaluation accuracy on development set of different models. As shown in the figure, the ensemble model that combines all three individual models performs best. AVG performs worst as a single model that even lower than the results of randomly guessing (randomly guessing achieves an accuracy 20% in a long run as the correct answer is uniformly distributed among all possible responses). A possible explain is that the model does not extract enough context-dependent information such as constitution and co-reference. In addition, although the difference between BERT and ensemble model is not significant, other two poorly performed models still provide useful information to the target task.

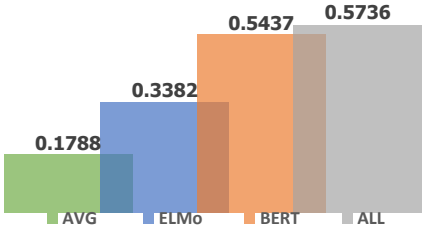

Figure 4: Accuracy on development set of single models and ensemble models

Figure 5 shows the evaluation results when we remove one of the component in the ensemble model. By comparing with Figure 4, it is easy to find the fact that when removing BERT from ensemble model, the overall performance decreases significantly. This result indicates that BERT extracts most amount of useful information from the input sequences, which may be benefit by the deep transformer model that

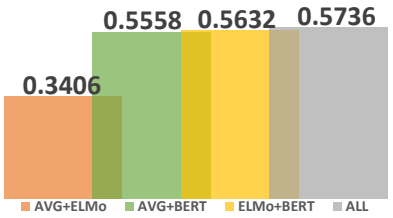

Figure 5: ensemble models accuracy on development set

learning context-dependent representations from large scale of pre-trained data. Another interesting fact is that, though AVG performs poorly as an individual model, it still increases the overall performance when it is combined with other models. This phenomenon indicates the effectiveness of ensemble approach. When combing features learned from different models, different representations learned from different methods and datasets is possible complementary to each other, which strengths the expression of complex information for the ensemble model. In addition, when combing to BERT, AVG+BERT and ELMo+BERT achieves similar performance, while AVG and ELMo differs obviously in individual performances. This result shows the possible conclusion that the amounts of complementary information from AVG and ELMo are similar. Although AVG is a context-independent model and ELMo is a context-dependent model, both of them are feature-based, which indicates the importance of fine-tuning on whole model in order to achieve better performance in downstream tasks.

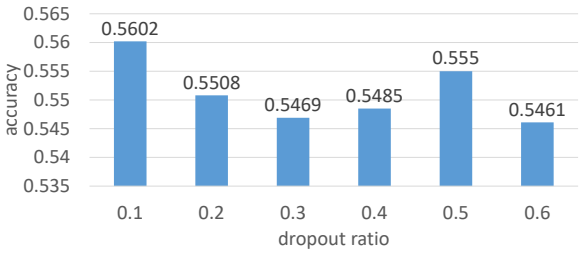

Figure 6: Accuracy of ensemble model w.r.t. dropout rate

Apart from the previous experiments, we also test the influence of dropout on the ensemble model. When treating the combined features as input to a normal feed forward network, dropout on the features can been seen as a random noise to the features. Figure 6 shows the result when changing the dropout rate from 0.1 to 0.6 with a step of 0.1. Except the function of avoiding over-fitting, we

can investigate the valid amount of features to the final performance. With the increasing of dropout rate, the performance slightly decreases, indicating that most of the features are valid in downstream task.

7 Conclusion & Future work

Under the support of a number of experiments data, combining context-dependent representations and context-independent representations of sentences as features to train on the downstream tasks will help model extract more useful information to do the downstream tasks. Our Ensemble models get improved comparing to single model, they have benefit such as comprehensive understanding of information from context and relatively fast training. However, it still have a large gap between human performance.

In our current work, we only test our models with randomly split data. To evaluate our model more precisely, we need to test them with question concept split data where each of the five sets have disjoint question concepts. Apart from this, more datasets or tasks will be included to give a more comprehensive evaluation.

With regard to our proposal, we planned to integrate knowledge from both ConceptNet (Speer et al., 2017) and GloVe, this idea drove us to extract vectors from ConcpetNet Numberbatch and trying to concatenated these vectors with intermediate layer vectors of our model. However, after this idea was implemented, we found that it does not enhance the performance of our model as we expected. The reason we think can explain this phenomenon is that the the CommonsenseQA dataset also generated part form ConcepNet, this means that using ConceptNet numberbatch could not provide extra knowledge for our initial model. Nevertheless, we believe that with some further feature engineering, the numberbatch could show its role on enhance the performance of our model. This can also be considered as our future work.

References

A. Ansari, M. Maknojia, and A. Shaikh. 2016. Intelligent question answering system based on artificial neural network. In *2016 IEEE International Conference on Engineering and Technology (ICETECH)*, pages 758–763.

Yllias Chali, Sadid A. Hasan, and Shafiq R. Joty. 2011. Improving graph-based random walks for complex question answering using syntactic, shallow semantic and extended string subsequence kernels. *Information Processing Management*, 47(6):843 – 855. Question Answering.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, and Phillipp Koehn. 2013. One billion word benchmark for measuring progress in statistical language modeling. *CoRR*, abs/1312.3005.

Christopher Clark and Matt Gardner. 2017. Simple and effective multi-paragraph reading comprehension. *arXiv preprint arXiv:1710.10723*.

J. Deng, W. Dong, R. Socher, L. Li, and and. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Sanjay K. Dwivedi and Vaishali Singh. 2013. Research and reviews in question answering system. *Procedia Technology*, 10:417 – 424. First International Conference on Computational Intelligence: Modeling Techniques and Applications (CIMTA) 2013.

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

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ankit Kumar, Ozan Irsoy, Jonathan Su, James Bradbury, Robert English, Brian Pierce, Peter Ondruska, Ishaan Gulrajani, and Richard Socher. 2015. Ask me anything: Dynamic memory networks for natural language processing. *CoRR*, abs/1506.07285.

S. P. Lende and M. M. Raghuwanshi. 2016. Question answering system on education acts using nlp techniques. In *2016 World Conference on Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave)*, pages 1–6.

Xiaodong Liu, Yelong Shen, Kevin Duh, and Jianfeng Gao. 2017. Stochastic answer networks for machine reading comprehension. *CoRR*, abs/1712.03556.

Xuezhe Ma and Eduard H. Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *CoRR*, abs/1603.01354.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *NIPS*.

Simon Ostermann, Michael Roth, Ashutosh Modi, Stefan Thater, and Manfred Pinkal. 2018. Semeval-2018 task 11: Machine comprehension using commonsense knowledge. In *Proceedings of The 12th*

International Workshop on Semantic Evaluation, pages 747–757. Association for Computational Linguistics.

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

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *CoRR*, abs/1802.05365.

Alec Radford. 2018. Improving language understanding by generative pre-training. OpenAI.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. *CoRR*, abs/1806.03822.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016a. Squad: 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250.

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

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge.

Sai Praneeth Suggu, Kushwanth Naga Goutham, Manoj K. Chinnakotla, and Manish Shrivastava. 2016. Hand in glove: Deep feature fusion network architectures for answer quality prediction in community question answering. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1429–1440. The COLING 2016 Organizing Committee.

Mihai Surdeanu and Christopher D Manning. 2010. Ensemble models for dependency parsing: cheap and good? In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 649–652. Association for Computational Linguistics.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *CoRR*, abs/1811.00937.

Ming Tan, Bing Xiang, and Bowen Zhou. 2015. Lstm-based deep learning models for non-factoid answer selection. *CoRR*, abs/1511.04108.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *BlackboxNLP@EMNLP*.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks? In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 3320–3328, Cambridge, MA, USA. MIT Press.

Lei Yu, Karl Moritz Hermann, Phil Blunsom, and Stephen Pulman. 2014. Deep learning for answer sentence selection. *CoRR*, abs/1412.1632.

Zhi-Hua Zhou. 2015. Ensemble learning. *Encyclopedia of biometrics*, pages 411–416.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. *CoRR*, abs/1506.06724.