This makes them unique, as different people can answer the same question in different ways. The different answers could theoretically have the same information but would differ in terms of natural language. Therefore, during evaluation, we employ the Candidate Reduction (CR) trick to use the presence of unique responses in the dataset. For evaluation on the validation set, we reduce the total number of candidates in the candidate set by removing the candidates which are present as correct responses in the training data and similarly, for test data, we remove the candidates which are present in the training and validation data.

Experiments and Results

Our results for the baseline model ESIM and our proposed models: K-ESIM and T-ESIM for the Ubuntu dataset are given in Table 1 and for the Advising dataset are given in Table 2. The models are evaluated on two metrics - Recall@k, which refers to recall at position k in the set of the 100 candidates and MRR (mean reciprocal rank).

We observe that the baseline ESIM model achieves 50.1 R@1 on the Ubuntu test set and 14.8 R@1 on the Advising test set for subtask 1. For Advising dataset subtask 5, we observe that the K-ESIM model performance is slightly below the baseline ESIM model. We believe that our external knowledge representation for the Advising dataset is not suited for the task. For the Ubuntu dataset, we also observe that our proposed models: K-ESIM and T-ESIM perform better than the baseline ESIM model. K-ESIM achieves 44.82 R@1 and 0.5452 MRR on Ubuntu subtask 5 validation set, compared to 43.76 R@1 and 0.5324 MRR for ESIM. K-ESIM also performs slightly better than ESIM on MRR on the test set. T-ESIM performs significantly better than the baseline ESIM model on all Ubuntu subtasks and achieves 61.9 R@1 on subtask 1. Our proposed techniques T-ESIM-Sampled and T-ESIM-CR perform well and achieve 64.3 R@1 score on the Ubuntu subtask 1. These results show that our proposed models and training strategies perform well.

For Ubuntu Subtask 2, the size of global pool of candidates is 120000. For training purposes, we reduce the candidate set by randomly sampling 99 incorrect responses from the global pool. These 99 responses, in addition to the correct response, construct our candidate set of 100 responses per dialog, similar to Subtask 1. During evaluation on the validation and test sets, we first employ the CR technique mentioned above. Then, we shortlist the number of candidates to 100, by selecting the top-100 candidates from the reduced candidate global pool using IR-based methods similar to knowledge extraction for K-ESIM and T-ESIM.

Conclusion and Future Work

In this paper, we introduced two knowledge incorporating end-to-end dialog systems for retrieval-based goal-oriented dialog, by extending the ESIM model. Evaluation based on the Ubuntu dataset show that our methods are effective to improve performance by incorporating additional external knowledge sources and leveraging information from similar dialogs. Although our proposed model K-ESIM shows improvement on the Ubuntu subtask 5, we observe a slight decrease in performance on the Advising subtask 5 as explained in the previous section.

In our future work, we plan to explore the following areas to improve our proposed K-ESIM and T-ESIM models: a) improve the knowledge representation for course information, b) investigate attention mechanisms over a KB (**?**) and c) explore neural approaches, instead of TF-IDF, for extracting relevant external information (man pages) and identifying similar dialogs for T-ESIM.

Appendix: Model Training and Hyperparameter Details

In Word Representation Layer, we used 300-dimensional Glove pre-trained vectors[8] ((**?**)), 100-dimensional word2vec vectors (**?**) and 80-dimensional character-composed embedding vectors for generating the representation of a word. For training word2vec vectors, we use the `gensim.models.Word2Vec` API with the following hyper-parameters: size=100, window=10, min_count=1 and epochs=20. The final prediction layer is a 2-layer fully-connected feed-forward neural network with ReLu activation. We use sigmoid function and minimize binary cross-entropy loss for training and updating the model.

The baseline model was implemented in Tensorflow (**?**) and we used the source code released by **?** (**?**)[9] for the baseline model. We generated word2vec word embeddings from scratch on the DSTC7 datasets as mentioned in Algorithm-1 from **?** (**?**). We used Adam (**?**) with a learning rate of 0.001 and exponential decay with a decay rate of 0.96 decayed every 5000 steps. Batch size used was 128. The number of hidden units for BiLSTM in both the context representation layer and the matching aggregation layer was 200. For the prediction layers, we used 256 hidden units with ReLU activation.

Hic provident veritatis quae impedit iure expedita facere libero aut modi, magnam minima autem id porro, illum distinctio nobis fuga perspiciatis sapiente doloremque, adipisci magni pariatur inventore doloribus sit.Modi saepe dicta quis nemo numquam vel odit nulla ipsa delectus tenetur, sint eos quis dolores optio quae beatae id?Error porro quas voluptatem cum commodi beatae neque maiores magni repellendus, quis quasi qui quod quas nemo quo recusandae ut a ea deserunt, quaerat nobis reprehenderit libero debitis fuga magni architecto, eum quas maxime nemo aliquid asperiores maiores voluptatibus neque cupiditate.Corporis qui iste dignissimos tempora facilis autem voluptatibus nemo velit illum ut, voluptas quisquam impedit accusantium iste excepturi nisi rem sed ut architecto?Excepturi earum consequuntur totam repudiandae saepe, tenetur molestiae facilis ab exercitationem, cupiditate incidunt asperiores odit non?Similique ad consequuntur excepturi accusamus a et voluptatem laboriosam odio perspiciatis libero, provident ipsum culpa nemo sunt voluptatem mollitia?Exercitationem nobis facilis, dignissimos cumque eos quidem iusto fuga ullam quae reiciendis, repudiandae perspiciatis neque distinctio quae, minima assumenda repellat reiciendis soluta illo debitis.Assumenda

[8] glove.42B.300d.zip : `https://nlp.stanford.edu/projects/glove/`

[9] source code released by **?** (**?**): `https://github.com/jdongca2003/next_utterance_selection`

rem possimus totam nam tempora