Iterative Multi-document Neural Attention for Multiple Answer Prediction

Claudio Greco
Department of Computer Science
University of Bari Aldo Moro
claudiogaetanogreco@gmail.com

Alessandro Suglia
Department of Computer Science
University of Bari Aldo Moro
alessandro.suglia@gmail.com

Pierpaolo Basile
Department of Computer Science
University of Bari Aldo Moro
pierpaolo.basile@uniba.it

Gaetano Rossiello
Department of Computer Science
University of Bari Aldo Moro
gaetano.rossiello@uniba.it

Giovanni Semeraro
Department of Computer Science
University of Bari Aldo Moro
giovanni.semeraro@uniba.it

Abstract

People have information needs of varying complexity, which can be solved by an intelligent agent able to answer questions formulated in a proper way, eventually considering user context and preferences. In a scenario in which the user profile can be considered as a question, intelligent agents able to answer questions can be used to find the most relevant answers for a given user. In this work we propose a novel model based on *Artificial Neural Networks* to answer questions with multiple answers by exploiting multiple facts retrieved from a knowledge base. The model is evaluated on the *factoid Question Answering* and *top-n recommendation* tasks of the *bAbI Movie Dialog* dataset. After assessing the performance of the model on both tasks, we try to define the long-term goal of a *conversational recommender system* able to interact using natural language and to support users in their information seeking processes in a personalized way.

1 Motivation and Background

We are surrounded by a huge variety of technological artifacts which "live" with us today. These artifacts can help us in several ways because they have the power to accomplish complex and time-consuming tasks. Unfortunately, common software systems can do for us only specific types of tasks, in a strictly algorithmic way which is pre-defined by the software designer. *Machine Learning (ML)*, a branch of *Artificial Intelligence (AI)*, gives machines the ability to learn to complete tasks without being explicitly programmed.

People have information needs of varying complexity, ranging from simple questions about common facts which can be found in encyclopedias, to more sophisticated cases in which they need to know what movie to watch during a romantic evening. These tasks can be solved by an intelligent agent able to answer questions formulated in a proper way, eventually considering user context and preferences.

Question Answering (QA) emerged in the last decade as one of the most promising fields in AI, since it allows to design intelligent systems which are able to give correct answers to user questions

expressed in natural language. Whereas, *recommender systems* produce individualized recommendations as output and have the effect of guiding the user in a personalized way to interesting or useful objects in a large space of possible options. In a scenario in which the user profile (the set of user preferences) can be represented by a question, intelligent agents able to answer questions can be used to find the most appealing items for a given user, which is the classical task that recommender systems can solve. Despite the efficacy of classical recommender systems, generally they are not able to handle a conversation with the user so they miss the possibility of understanding his contextual information, emotions and feedback to refine the user profile and provide enhanced suggestions. *Conversational recommender systems* assist online users in their information-seeking and decision making tasks by supporting an interactive process [10] which could be goal oriented with the task of starting general and, through a series of interaction cycles, narrowing down the user interests until the desired item is obtained [17].

In this work we propose a novel model based on *Artificial Neural Networks* to answer questions exploiting multiple facts retrieved from a knowledge base and evaluate it on a *QA* task. Moreover, the effectiveness of the model is evaluated on the *top-n recommendation* task, where the aim of the system is to produce a list of suggestions ranked according to the user preferences. After having assessed the performance of the model on both tasks, we try to define the long-term goal of a *conversational recommender system* able to interact with the user using natural language and to support him in the information seeking process in a personalized way.

In order to fulfill our long-term goal of building a *conversational recommender system* we need to assess the performance of our model on specific tasks involved in this scenario. A recent work which goes in this direction is reported in [6], which presents the *bAbI Movie Dialog* dataset, composed by different tasks such as *factoid QA*, *top-n recommendation* and two more complex tasks, one which mixes *QA* and recommendation and one which contains turns of dialogs taken from Reddit. Having more specific tasks like *QA* and recommendation, and a more complex one which mixes both tasks gives us the possibility to evaluate our model on different levels of granularity. Moreover, the subdivision in turns of the more complex task provides a proper benchmark of the model capability to handle an effective dialog with the user.

For the task related to *QA*, a lot of datasets have been released in order to assess the machine reading and comprehension capabilities and a lot of neural network-based models have been proposed. Our model takes inspiration from [19], which is able to answer *Cloze-style* [22] questions repeating an attention mechanism over the query and the documents multiple times. Despite the effectiveness on the *Cloze-style* task, the original model does not consider multiple documents as a source of information to answer questions, which is fundamental in order to extract the answer from different relevant facts. The restricted assumption that the answer is contained in the given document does not allow the model to provide an answer which does not belong to the document. Moreover, this kind of task does not expect multiple answers for a given question, which is important for the complex information needs required for a *conversational recommender system*.

According to our vision, the main outcomes of our work can be considered as building blocks for a *conversational recommender system* and can be summarized as follows:

1. we extend the model reported in [19] to let the inference process exploit evidences observed in multiple documents coming from an external knowledge base represented as a collection of textual documents;

2. we design a model able to leverage the attention weights generated by the inference process to provide multiple answers which does not necessarily belong to the documents through a multi-layer neural network which may uncover possible relationships between the most relevant evidences;

3. we assess the efficacy of our model through an experimental evaluation on *factoid QA* and *top-n recommendation* tasks supporting our hypothesis that a *QA* model can be used to solve *top-n recommendation*, too.

The paper is organized as follows: Section 2 describes our model, while Section 3 summarizes the evaluation of the model on the two above-mentioned tasks and the comparison with respect to state-of-the-art approaches. Section 4 gives an overview of the literature of both *QA* and recommender systems, while final remarks and our long-term vision are reported in Section 5.

2 Methodology

Given a query q, an operator $\psi : Q \rightarrow D$ that produces the set of documents relevant for q, where Q is the set of all queries and D is the set of all documents. Our model defines a workflow in which a sequence of inference steps are performed in order to extract relevant information from $\psi(q)$ to generate the answers for q.

Following [19], our workflow consists of three steps: (1) the *encoding* phase, which generates meaningful representations for query and documents; (2) the *inference* phase, which extracts relevant semantic relationships between the query and the documents by using an iterative attention mechanism and finally (3) the *prediction* phase, which generates a score for each candidate answer.

2.1 Encoding phase

The input of the encoding phase is given by a query q and a set of documents $\psi(q) = \{d_1, d_2, \ldots, d_{|D_q|}\} \equiv D_q$. Both queries and documents are represented by a sequence of words $X = (x_1, x_2, \ldots, x_{|X|})$, drawn from a vocabulary V. Each word is represented by a continuous d-dimensional word embedding $\mathbf{x} \in \mathbb{R}^d$ stored in a word embedding matrix $\mathbf{X} \in \mathbb{R}^{|V| \times d}$.

The sequences of dense representations for q and d_j are encoded using a *bidirectional recurrent neural network encoder* with *Gated Recurrent Units (GRU)* as in [19] which represents each word $x_i \in X$ as the concatenation of a forward encoding $\overrightarrow{\mathbf{h}_k} \in \mathbb{R}^h$ and a backward encoding $\overleftarrow{\mathbf{h}_k} \in \mathbb{R}^h$. From now on, we denote the contextual representation for the word q_i by $\tilde{\mathbf{q}}_i \in \mathbb{R}^{2h}$ and the contextual representation for the word $d_{j,i}$ in the document d_j by $\tilde{\mathbf{d}}_{j,i} \in \mathbb{R}^{2h}$. Differently from [19], we build a unique representation for the whole set of documents D_q related to the query q by stacking each contextual representation $\tilde{\mathbf{d}}_{j,i}$ obtaining a matrix $\tilde{\mathbf{D}}_q \in \mathbb{R}^{l \times 2h}$, where $l = |d_1| + |d_2| + \ldots + |d_{|D_q|}|$.

2.2 Inference phase

This phase uncovers a possible inference chain which models meaningful relationships between the query and the set of related documents. The inference chain is obtained by performing, for each inference step $t = 1, 2, \ldots, T$, the attention mechanisms given by the *query attentive read* and the *document attentive read* keeping a state of the inference process given by an additional *recurrent neural network* with *GRU* units. In this way, the network is able to progressively refine the attention weights focusing on the most relevant tokens of the query and the documents which are exploited by the prediction neural network to select the correct answers among the candidate ones.

2.2.1 Query attentive read

Given the contextual representations for the query words $(\tilde{\mathbf{q}}_1, \tilde{\mathbf{q}}_2, \ldots, \tilde{\mathbf{q}}_{|q|})$ and the inference *GRU* state $\mathbf{s}_{t-1} \in \mathbb{R}^s$, we obtain a refined query representation $\mathbf{q}_t$ (*query glimpse*) by performing an attention mechanism over the query at inference step t:

$$\hat{q}_{i,t} = \underset{i=1,\ldots,|q|}{\text{softmax}}\, \tilde{\mathbf{q}}_i^\top (\mathbf{A}_q \mathbf{s}_{t-1} + \mathbf{a}_q),$$

$$\mathbf{q}_t = \sum_i \hat{q}_{i,t} \tilde{\mathbf{q}}_i$$

where $\hat{q}_{i,t}$ are the attention weights associated to the query words, $\mathbf{A}_q \in \mathbb{R}^{2h \times s}$ and $\mathbf{a}_q \in \mathbb{R}^{2h}$ are respectively a weight matrix and a bias vector which are used to perform the bilinear product with the query token representations $\tilde{\mathbf{q}}_i$. The attention weights can be interpreted as the relevance scores for each word of the query dependent on the inference state s_{t-1} at the current inference step t.

2.2.2 Document attentive read

Given the query glimpse $\mathbf{q}_t$ and the inference *GRU* state $\mathbf{s}_{t-1} \in \mathbb{R}^s$, we perform an attention mechanism over the contextual representations for the words of the stacked documents $\tilde{\mathbf{D}}_q$:

$$\hat{d}_{i,t} = \underset{i=1,\dots,l}{\text{softmax}}\, \tilde{\mathbf{D}}_{q_i}^\top (\mathbf{A}_d[\mathbf{s}_{t-1}, \mathbf{q}_t] + \mathbf{a}_d),$$

$$\mathbf{d}_t = \sum_i \hat{d}_{i,t} \tilde{\mathbf{D}}_{q_i}$$

where $\tilde{\mathbf{D}}_{q_i}$ is the i-th row of $\tilde{\mathbf{D}}_q$, $\hat{d}_{i,t}$ are the attention weights associated to the document words, $\mathbf{A}_d \in \mathbb{R}^{2h \times s}$ and $\mathbf{a}_d \in \mathbb{R}^{2h}$ are respectively a weight matrix and a bias vector which are used to perform the bilinear product with the document token representations $\tilde{\mathbf{D}}_{q_i}$. The attention weights can be interpreted as the relevance scores for each word of the documents conditioned on both the query glimpse and the inference state s_{t-1} at the current inference step t. By combining the set of relevant documents in $\tilde{\mathbf{D}}_q$, we obtain the probability distribution $(\hat{d}_{1,t}, \hat{d}_{2,t}, \dots \hat{d}_{l,t})$ over all the relevant document tokens using the above-mentioned attention mechanism.

2.2.3 Gating search results

The inference *GRU* state at the inference step t is updated according to $\mathbf{s}_t = GRU([\mathbf{r}_q \cdot \mathbf{q}_t, \mathbf{r}_d \cdot \mathbf{d}_t], \mathbf{s}_{t-1})$, where r_q and r_d are the results of a gating mechanism obtained by evaluating $g([\mathbf{s}_{t-1}, \mathbf{q}_t, \mathbf{d}_t, \mathbf{q}_t \cdot \mathbf{d}_t])$ for the query and the documents, respectively. The gating function $g : \mathbb{R}^{s+6h} \rightarrow \mathbb{R}^{2h}$ is defined as a 2-layer feed-forward neural network with a *Rectified Linear Unit (ReLU)* [12] activation function in the hidden layer and a *sigmoid* activation function in the output layer. The purpose of the gating mechanism is to retain useful information for the inference process about query and documents and forget useless one.

2.3 Prediction phase

The prediction phase, which is completely different from the *pointer-sum* loss reported in [19], is able to generate, given the query q, a relevance score for each candidate answer $a \in A$ by using the document attention weights $\hat{d}_{i,T}$ computed in the last inference step T. The relevance score of each word w is obtained by summing the attention weights of w in each document related to q. Formally the relevance score for a given word w is defined as:

$$score(w) = \frac{1}{\pi(w)} \sum_{i=1}^{l} \phi(i, w)$$

where $\phi(i, w)$ returns 0 if $\sigma(i) \neq w$, $\hat{d}_{i,T}$ otherwise; $\sigma(i)$ returns the word in position i of the stacked documents matrix $\tilde{\mathbf{D}}_q$ and $\pi(w)$ returns the frequency of the word w in the documents D_q related to the query q. The relevance score takes into account the importance of token occurrences in the considered documents given by the computed attention weights. Moreover, the normalization term $\frac{1}{\pi(w)}$ is applied to the relevance score in order to mitigate the weight associated to highly frequent tokens.

The evaluated relevance scores are concatenated in a single vector representation $\mathbf{z} = [score(w_1), score(w_2), \dots, score(w_{|V|})]$ which is given in input to the answer prediction neural network defined as:

$$\mathbf{y} = \text{sigmoid}(\mathbf{W}_{ho}\, \text{relu}(\mathbf{W}_{ih}\mathbf{z} + \mathbf{b}_{ih}) + \mathbf{b}_{ho})$$

where u is the hidden layer size, $\mathbf{W}_{ih} \in \mathbb{R}^{u \times |V|}$ and $\mathbf{W}_{ho} \in \mathbb{R}^{|A| \times u}$ are weight matrices, $\mathbf{b}_{ih} \in \mathbb{R}^u$, $\mathbf{b}_{ho} \in \mathbb{R}^{|A|}$ are bias vectors, sigmoid(x) $= \frac{1}{1+e^{-x}}$ is the *sigmoid* function and relu(x) $= max(0, x)$ is the *ReLU* activation function, which are applied pointwise to the given input vector.

The neural network weights are supposed to learn latent features which encode relationships between the most relevant words for the given query to predict the correct answers. The outer *sigmoid* activation function is used to treat the problem as a *multi-label* classification problem, so that each candidate answer is independent and not mutually exclusive. In this way the neural network

generates a score which represents the probability that the candidate answer is correct. Moreover, differently from [19], the candidate answer A can be any word, even those which not belong to the documents related to the query.

The model is trained by minimizing the *binary cross-entropy* loss function comparing the neural network output y with the target answers for the given query q represented as a binary vector, in which there is a 1 in the corresponding position of the correct answer, 0 otherwise.

3 Experimental evaluation

The model performance is evaluated on the *QA* and *Recs* tasks of the *bAbI Movie Dialog* dataset using *HITS@k* evaluation metric, which is equal to the number of correct answers in the top-k results. In particular, the performance for the *QA* task is evaluated according to *HITS@1*, while the performance for the *Recs* task is evaluated according to *HITS@100*.

Differently from [6], the relevant knowledge base facts, taken from the knowledge base in triple form distributed with the dataset, are retrieved by ψ implemented by exploiting the *Elasticsearch* engine and not according to an hash lookup operator which applies a strict filtering procedure based on word frequency. In our work, ψ returns at most the top 30 relevant facts for q. Each entity in questions and documents is recognized using the list of entities provided with the dataset and considered as a single word of the dictionary V.

Questions, answers and documents given in input to the model are preprocessed using the *NLTK* toolkit [2] performing only word tokenization. The question given in input to the ψ operator is preprocessed performing word tokenization and stopword removal.

The optimization method and tricks are adopted from [19]. The model is trained using *ADAM* [9] optimizer (*learning rate*=0.001) with a batch size of 128 for at most 100 epochs considering the best model until the *HITS@k* on the validation set decreases for 5 consecutive times. *Dropout* [20] is applied on r_q and on r_d with a rate of 0.2 and on the prediction neural network hidden layer with a rate of 0.5. *L2 regularization* is applied to the embedding matrix $\mathbf{X}$ with a coefficient equal to 0.0001. We clipped the gradients if their norm is greater than 5 to stabilize learning [14]. Embedding size d is fixed to 50. All *GRU* output sizes are fixed to 128. The number of inference steps T is set to 3. The size of the prediction neural network hidden layer u is fixed to 4096. Biases $\mathbf{b}_{ih}$ and $\mathbf{b}_{ho}$ are initialized to zero vectors. All weight matrices are initialized sampling from the normal distribution $\mathcal{N}(0, 0.05)$. The *ReLU* activation function in the prediction neural network has been experimentally chosen comparing different activation functions such as *sigmoid* and *tanh* and taking the one which leads to the best performance. The model is implemented in *TensorFlow* [1] and executed on an *NVIDIA TITAN X* GPU.

METHODS	QA TASK	RECS TASK
QA SYSTEM	**90.7**	N/A
SVD	N/A	19.2
LSTM	6.5	27.1
SUPERVISED EMBEDDINGS	50.9	29.2
MEMN2N	79.3	28.6
JOINT SUPERVISED EMBEDDINGS	43.6	28.1
JOINT MEMN2N	83.5	26.5
OUR MODEL	86.8	**30.0**

Table 1: Comparison between our model and baselines from [6] on the *QA* and *Recs* tasks evaluated according to *HITS@1* and *HITS@100*, respectively.

Following the experimental design, the results in Table 1 are promising because our model outperforms all other systems on both tasks except for the *QA SYSTEM* on the *QA* task. Despite the advantage of the *QA SYSTEM*, it is a carefully designed system to handle knowledge base data in the form of triples, but our model can leverage data in the form of documents, without making any assumption about the form of the input data and can be applied to different kind of tasks. Additionally, the model *MEMN2N* is a neural network whose weights are pre-trained on the same dataset without using the long-term memory and the models *JOINT SUPERVISED EMBEDDINGS* and *JOINT MEMN2N* are models trained across all the tasks of the dataset in order to boost performance. De-

spite that, our model outperforms the three above-mentioned ones without using any supplementary trick. Even though our model performance is higher than all the others on the *Recs* task, we believe that the obtained result may be improved and so we plan a further investigation. Moreover, the need for further investigation can be justified by the work reported in [18] which describes some issues regarding the *Recs* task.

Figure 1 shows the attention weights computed in the last inference step of the iterative attention mechanism used by the model to answer to a given question. Attention weights, represented as red boxes with variable color shades around the tokens, can be used to interpret the reasoning mechanism applied by the model because higher shades of red are associated to more relevant tokens on which the model focus its attention. It is worth to notice that the attention weights associated to each token are the result of the inference mechanism uncovered by the model which progressively tries to focus on the relevant aspects of the query and the documents which are exploited to generate the answers.

Figure 1: Attention weights $\tilde{\mathbf{q}}_i$ and $\tilde{\mathbf{D}}_{q_i}$ computed by the neural network attention mechanisms at the last inference step T for each token. Higher shades correspond to higher relevance scores for the related tokens.

Given the question "what does Larenz Tate act in?" shown in the above-mentioned figure, the model is able to understand that "Larenz Tate" is the subject of the question and "act in" represents the intent of the question. Reading the related documents, the model associates higher attention weights to the most relevant tokens needed to answer the question, such as "The Postman", "A Man Apart" and so on.

4 Related work

We think that it is necessary to consider models and techniques coming from research both in *QA* and recommender systems in order to pursue our desire to build an intelligent agent able to assist the user in decision-making tasks. We cannot fill the gap between the above-mentioned research areas if we do not consider the proposed models in a synergic way by virtue of the proposed analogy between the user profile (the set of user preferences) and the items to be recommended, as the question and the correct answers. The first work which goes in this direction is reported in [11], which exploits movie descriptions to suggest appealing movies for a given user using an architecture tipically used for *QA* tasks. In fact, most of the research in the recommender systems field presents ad-hoc systems which exploit neighbourhood information like in *Collaborative Filtering* techniques [13], item descriptions and metadata like in *Content-based* systems [5]. Recently presented neural network models [4, 3] systems are able to learn latent representations in the network weights leveraging information coming from user preferences and item information.

In recent days, a lot of effort is devoted to create benchmarks for artificial agents to assess their ability to comprehend natural language and to reason over facts. One of the first attempt is the *bAbI* [24] dataset which is a synthetic dataset containing elementary tasks such as selecting an answer between one or more candidate facts, answering yes/no questions, counting operations over lists and sets and basic induction and deduction tasks. Another relevant benchmark is the one described in [7], which provides *CNN/Daily Mail* datasets consisting of document-query-answer triples where an entity in the query is replaced by a placeholder and the system should identify the correct entity by

reading and comprehending the given document. *MCTest* [16] requires machines to answer multiple-choice reading comprehension questions about fictional stories, directly tackling the high-level goal of open-domain machine comprehension. Finally, *SQuAD* [15] consists in a set of *Wikipedia* articles, where the answer to each question is a segment of text from the corresponding reading passage.

According to the experimental evaluations conducted on the above-mentioned datasets, high-level performance can be obtained exploiting complex attention mechanisms which are able to focus on relevant evidences in the processed content. One of the earlier approaches used to solve these tasks is given by the general *Memory Network* [23, 21] framework which is one of the first neural network models able to access external memories to extract relevant information through an attention mechanism and to use them to provide the correct answer. A deep *Recurrent Neural Network* with *Long Short-Term Memory* units is presented in [7], which solves *CNN/Daily Mail* datasets by designing two different attention mechanisms called *Impatient Reader* and *Attentive Reader*. Another way to incorporate attention in neural network models is proposed in [8] which defines a *pointer-sum* loss whose aim is to maximize the attention weights which lead to the correct answer.

5 Conclusions and Future Work

In this work we propose a novel model based on *Artificial Neural Networks* to answer questions with multiple answers by exploiting multiple facts retrieved from a knowledge base. The proposed model can be considered a relevant building block of a *conversational recommender system*. Differently from [19], our model can consider multiple documents as a source of information in order to generate multiple answers which may not belong to the documents. As presented in this work, common tasks such as *QA* and *top-n recommendation* can be solved effectively by our model.

In a common recommendation system scenario, when a user enters a search query, it is assumed that his preferences are known. This is a stringent requirement because users cannot have a clear idea of their preferences at that point. Conversational recommender systems support users to fulfill their information needs through an interactive process. In this way, the system can provide a personalized experience dynamically adapting the user model with the possibility to enhance the generated predictions. Moreover, the system capability can be further enhanced giving explanations to the user about the given suggestions.

To reach our goal, we should improve our model by designing a ψ operator able to return relevant facts recognizing the most relevant information in the query, by exploiting user preferences and contextual information to learn the user model and by providing a mechanism which leverages attention weights to give explanations. In order to effectively train our model, we plan to collect real dialog data containing contextual information associated to each user and feedback for each dialog which represents if the user is satisfied with the conversation. Given these enhancements, we should design a system able to hold effectively a dialog with the user recognizing his intent and providing him the most suitable contents.

With this work we try to show the effectiveness of our architecture for tasks which go from *pure question answering* to *top-n recommendation* through an experimental evaluation without any assumption on the task to be solved. To do that, we do not use any hand-crafted linguistic features but we let the system learn and leverage them in the inference process which leads to the answers through multiple reasoning steps. During these steps, the system understands relevant relationships between question and documents without relying on canonical matching, but repeating an attention mechanism able to unconver related aspects in distributed representations, conditioned on an encoding of the inference process given by another neural network. Equipping agents with a reasoning mechanism like the one described in this work and exploiting the ability of neural network models to learn from data, we may be able to create truly intelligent agents.

6 Acknowledgments

This work is supported by the *IBM Faculty Award "Deep Learning to boost Cognitive Question Answering"*. The Titan X GPU used for this research was donated by the *NVIDIA Corporation*.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. J. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Józefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. G. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. A. Tucker, V. Vanhoucke, V. Vasudevan, F. B. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *CoRR*, abs/1603.04467, 2016.

[2] S. Bird. Nltk: the natural language toolkit. In *Proceedings of the COLING/ACL on Interactive presentation sessions*, pages 69–72. Association for Computational Linguistics, 2006.

[3] H. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, R. Anil, Z. Haque, L. Hong, V. Jain, X. Liu, and H. Shah. Wide & deep learning for recommender systems. *CoRR*, abs/1606.07792, 2016.

[4] P. Covington, J. Adams, and E. Sargin. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*, New York, NY, USA, 2016.

[5] M. de Gemmis, P. Lops, C. Musto, F. Narducci, and G. Semeraro. Semantics-aware content-based recommender systems. In *Recommender Systems Handbook*, pages 119–159. Springer, 2015.

[6] J. Dodge, A. Gane, X. Zhang, A. Bordes, S. Chopra, A. Miller, A. Szlam, and J. Weston. Evaluating prerequisite qualities for learning end-to-end dialog systems. *arXiv preprint arXiv:1511.06931*, 2015.

[7] K. M. Hermann, T. Kociský, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701, 2015.

[8] R. Kadlec, M. Schmid, O. Bajgar, and J. Kleindienst. Text understanding with the attention sum reader network. *arXiv preprint arXiv:1603.01547*, 2016.

[9] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[10] T. Mahmood and F. Ricci. Improving recommender systems with adaptive conversational strategies. In *Proceedings of the 20th ACM conference on Hypertext and hypermedia*, pages 73–82. ACM, 2009.

[11] C. Musto, C. Greco, A. Suglia, and G. Semeraro. Ask me any rating: A content-based recommender system based on recurrent neural networks. In *Proceedings of the 7th Italian Information Retrieval Workshop, Venezia, Italy, May 30-31, 2016.*

[12] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010.

[13] X. Ning, C. Desrosiers, and G. Karypis. A comprehensive survey of neighborhood-based recommendation methods. In *Recommender Systems Handbook*, pages 37–76. Springer, 2015.

[14] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. *ICML (3)*, 28:1310–1318, 2013.

[15] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. Squad: 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250, 2016.

[16] M. Richardson, C. J. C. Burges, and E. Renshaw. Mctest: A challenge dataset for the open-domain machine comprehension of text. In *EMNLP*, 2013.

[17] N. Rubens, D. Kaplan, and M. Sugiyama. Active learning in recommender systems. In *Recommender Systems Handbook*, pages 809–846. Springer, 2015.

[18] R. Searle and M. Bingham-Walker. Why "blow out"? a structural analysis of the movie dialog dataset. *ACL 2016*, page 215, 2016.

[19] A. Sordoni, P. Bachman, and Y. Bengio. Iterative alternating neural attention for machine reading. *arXiv preprint arXiv:1606.02245*, 2016.

[20] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[21] S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448, 2015.

[22] W. L. Taylor. Cloze procedure: a new tool for measuring readability. *Journalism and Mass Communication Quarterly*, 30(4):415, 1953.

[23] J. Weston, S. Chopra, and A. Bordes. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014.

[24] J. Weston, A. Bordes, S. Chopra, and T. Mikolov. Towards ai-complete question answering: A set of prerequisite toy tasks. *CoRR*, abs/1502.05698, 2015.