Adversarial Networks for Machine Reading

Quentin Grail — Julien Perez — Tomi Silander

NAVER LABS Europe,
6-8, chemin de Maupertuis, 38240 Meylan, France

ABSTRACT. *Deep machine reading models have recently progressed remarkably with the help of differentiable reasoning models. In this context, deep end-to-end trainable networks enhanced with memory and attention have demonstrated promising performance on simple natural language based reasoning tasks. However, the training of machine comprehension models commonly requires a large annotated question-answer dataset for learning. In this paper, we explore the paradigm of adversarial learning and self-play for machine reading comprehension. Inspired by the success in the domain of game learning, we propose a novel approach to train machine comprehension models based on a coupled attention-based model. In this approach, a reader network is in charge of finding answers to the questions regarding a passage of text, while an obfuscation network tries to obfuscate spans of text in order to minimize the probability of success of the reader. The model is evaluated on several question-answering corpora. The proposed learning paradigm and associated models show promising results.*

RÉSUMÉ. *Les modèles d'apprentissage profond utilisés sur des tâches de lecture automatique ont remarquablement progressé ces dernières années. Parmi ces architectures, les modèles d'attention et à mémoire ont démontré des performances encourageantes sur différentes tâches de raisonnement. Cependant, le protocole d'apprentissage de ces modèles suppose qu'une grande quantité d'exemples soient disponibles. Dans cet article, nous proposons d'exploiter l'apprentissage adversarial et le self-play durant la phase d'entraînement des modèles. Nous proposons un nouveau protocole d'apprentissage sous la forme d'un jeu entre deux modèles adverses. Ces modèles sont mis en compétition sur une tâche de question-réponse. D'un côté un modèle, appelé « lecteur », est chargé de répondre à une question portant sur un document, et de l'autre un modèle, appelé « réseau d'obfuscation », est chargé d'obfusquer un passage du document de manière à maximiser la probabilité de tromper le lecteur sur ce document corrompu. Nous avons testé ce protocole sur plusieurs datasets de question-réponse, et ce nouveau protocole d'apprentissage adversarial permet d'obtenir des résultats encourageants.*

KEYWORDS: *Machine reading, adversarial learning, deep learning.*

MOTS-CLÉS : *Modèles de lecture, apprentissage adversarial, apprentissage profond.*

1. Introduction

Automatic comprehension of text is one of the main goals of natural language processing. While the ability of a machine to understand text can be assessed in many different ways, several benchmark datasets have recently been created to focus on answering questions as a way to evaluate machine comprehension (Richardson *et al.*, 2013; Hermann *et al.*, 2015; Hill *et al.*, 2015; Weston *et al.*, 2015; Rajpurkar *et al.*, 2016; Nguyen *et al.*, 2016). In this setup, a piece of text such as a news article or a story is presented to the machine. The machine is then expected to answer one or multiple questions related to the text. Figure1 presents question-answering example from the Cambridge dataset. Solving this task provides tools that help users to efficiently access large amounts of information. Furthermore, it also acts as an important proxy task to assess models of natural language understanding and reasoning. Recently, publication of many large datasets (Hermann *et al.*, 2015; Rajpurkar *et al.*, 2016; Trischler *et al.*, 2017; Nguyen *et al.*, 2016) have contributed to significant advancement in machine comprehension and question-answering. Recent neural models for machine comprehension are now approaching human comprehension on some of these benchmarks, and there is currently a lot of novel and promising research on parametric models that feature reasoning capabilities using techniques such as attention and memory. The work in this field is currently following the paradigm of supervised learning which makes it strictly dependent on the availability of annotated datasets; the production of which is costly. Since the 1990s an increasingly common research activity has been dedicated to self-play and adversariality to overcome this dependency and allow a model to exploit its own decisions to improve itself. Some famous examples are related to policy learning in games. TD-Gammon (Tesauro, 1995) was a neural network controller for backgammon which achieved near top player performance using self-play as learning paradigm. More recently, DeepMind's AlphaGo (Silver *et al.*, 2016) used the same paradigm to win against the currently best human Go player. The major advantage of such a setting is to alleviate the learning procedure's dependency on an available annotated dataset. Two models can be set up to learn and improve their performance by acting one against the other in so-called sparring patterns.

In this paper, we adapt this paradigm to the domain of machine reading. On the first hand, a **reader network** is trained to learn to answer questions regarding a passage of text. On the other hand, an **obfuscation network** learns to obfuscate words of a given passage in order to minimize the probability of the reading model to successfully answer the question. We developed a sequential learning protocol in order to gradually improve the quality of the models. This paradigm separates itself from the current approach of joint question and answer learning from text as proposed by Wang *et al.* (2017a). Indeed, rather than using question generation as regularizer of a reader model, we suggest using adversarial training to free us from the constraint of strict and bounded supervision and to enhance the robustness of the answering model.

Our contributions can be summarized as follows: (1) We propose a new learning paradigm for machine comprehension based on adversarial training. (2) With exper-

Document: *This is a very reasonably priced hotel of a good standard for a short visit. I had a single room (407) at the front of the hotel. On the negative side, the room was very small, there is street and aircraft noise and it was too warm. However, they have made excellent use of the space available and the decor was good. The bathroom was newly fitted out and the shower was excellent. I found the staff efficient and friendly. Much better than the last place I stayed in London and cheaper. I didn't have breakfast but it was reasonably priced.*
Question: *How is the service?*
Answer: *3/5.*

Figure 1. *An example from the TripAdvisor dataset.*

iments in several machine reading corpora and with several neural architectures, we show that this methodology allows us to overcome the requirement of strict supervision and provides robustness to noise in question answering. (3) The attention mechanism allows the visualization of passages considered as meaningful by the obfuscation network. We present the results of this attention mechanism on multiple examples.

Roadmap:
In Section 2 we review the state-of-the-art of machine reading comprehension, the paradigm of adversarial learning and its relation to the adversarial learning protocol proposed in this article. In Section 3, we formalize our adversarial learning protocol and introduce the two types of architectures used in this work. In Section 4 we introduce the corpora used for evaluation. In Section 5 we present our experimental results, and finally in Section 6, we demonstrate several visualizations of the decisions and attention values produced by the coupled models.

2. Related work

2.1. *End-to-end machine reading*

The task of end-to-end machine reading consists of learning, in a supervised manner, to answer a question given a passage of text. One of the popular formal settings of the problem is the cloze-style question-answering task. This task involves tuples of the form (d, q, a, C), where d is a document (context) and q is a query over the contents of d, in which a word has been replaced with a placeholder. The objective is to fill the placeholder with a word chosen among the set of candidates C. The correct answer is a. In this work, we consider datasets where each candidate $c \in C$ has at least one token which also appears in the document. The task can then be described as: given a document-query pair (d, q), find $a \in C$ which answers q. Below we provide an overview of representative neural network architectures which have been applied to this problem.

LSTMs with Attention: Several architectures introduced in Hermann *et al.* (2015) employ LSTM units to compute a combined document-query representation $g(d, q)$, which is used to rank the candidate answers. These include the DeepLSTM Reader which performs a single forward pass through the concatenated (document, query) pair to obtain $g(d, q)$; the Attentive Reader which first computes a document vector $d(q)$ by a weighted aggregation of words according to attentions based on q, and then combines $d(q)$ and q to obtain their joint representation $g(d(q), q)$; and the Impatient Reader where the document representation is built incrementally. The architecture of the Attentive Reader has been simplified recently in Stanford Attentive Reader, where shallower recurrent units were used with a bilinear form for the query-document attention (Chen *et al.*, 2016).

Attention-Sum Reader: The Attention-Sum (AS) Reader (Kadlec *et al.*, 2016) uses two bidirectional GRU networks to encode both d and q into vectors. A probability distribution over the entities in d is obtained by computing dot products between q and the entity embeddings and taking the softmax. Then, an aggregation scheme called "pointer-sum attention" is further applied to sum the probabilities of the same entity, so that frequent entities in the document will be favored compared to rare ones. Building on the AS Reader, the Attention-over-Attention (AoA) Reader (Cui *et al.*, 2017) introduces a two-way attention mechanism where the query and the document are mutually attentive to each other.

Multi-hop Architectures: Memory Networks (MemNets) were proposed in Weston *et al.* (2014), where each sentence in the document is encoded to a memory cell by aggregating nearby words. Attention over the memory slots given the query is used to compute an overall attention and to renew the query representation over multiple iterations, allowing certain types of reasoning over the salient facts in the memory and the query. Neural Semantic Encoders (NSE) (Yu and Munkhdalai, 2017) extended MemNets by introducing a write operation which can evolve the memory over time during the course of reading. Iterative reasoning has been found effective in several more recent models, including the Iterative Attentive Reader (Sordoni *et al.*, 2016) and ReasoNet (Shen *et al.*, 2016). The latter allows dynamic reasoning steps and is trained with reinforcement learning.

In other related work, EpiReader (Trischler *et al.*, 2016) consists of two networks, where one proposes a small set of candidate answers, and the other reranks the proposed candidates conditioned on the query and the context. Bi-Directional Attention Flow network (BiDAF) (Seo *et al.*, 2016) adopts a multi-stage hierarchical architecture along with a flow-based attention mechanism.

2.2. *Adversarial learning*

The idea of using an adversarial learning protocol has been very popular during the last couple of years, particularly in the field of generative models. Indeed Generative Adversarial Networks (GANs), introduced in (Goodfellow *et al.*, 2014a), have now

lots of applications and allowed the training protocol to go beyond the strict supervision of the answer. The main principle of Generative Adversarial Networks (GANs) is to train jointly two adversarial models. These two models are challenging each other with opposing objectives and jointly progressing in the task they are designed for. In machine reading, it has been recently observed that answering a question regarding a text passage and predicting the question regarding a text passage are interesting tasks to model jointly. Consequently, several papers have proposed using the question generation as a regularization task to improve the passage encoding model of a neural reader (Yuan *et al.*, 2017; Wang *et al.*, 2017a). In this paper, we acknowledge that these two tasks may indeed be complementary but we believe adversarial training in two player games will lead to similar advantages than those observed previously. As generating a question for a passage is hard, we adapt recent work by Guo *et al.* (2017) and define the learning of an obfuscation network as a complementary task to the task of learning a reader. Such an obfuscation network tries to find the most meaningful spans of text to obfuscate in a given passage for a given question in order to minimize the probability of the reader successfully answering the question.

2.3. *Adaptive dropout*

Several studies have recently featured the idea of challenging deep machine reading models with adversarial examples (Miyato *et al.*, 2016; Jia and Liang, 2017). While this kind of approach is well known in computer vision (Goodfellow *et al.*, 2014b), it seems to be relevant also for natural language processing. More precisely, Jia and Liang (2017) demonstrated that a large majority of the recent state-of-the-art deep machine reading models suffer from a lack of robustness regarding adversarial examples because of their oversensitivity. It means that small perturbation in the input can completely disturb the model. In these studies, models suffer from the so-called *catastrophic forgetting*; their average accuracies were decreased by half when tested on corrupted data, i.e., on documents with an additional sentence at the end, which normally should not affect the answer.

One of the attempts to prevent overfitting is to randomly drop network units while training (Srivastava *et al.*, 2014). Such an approach effectively results in combining many different neural networks to make a prediction. In the same spirit, training a model on a dataset with corrupted data is shown to decrease overfitting. Maaten *et al.* (2013) suggest different ways to corrupt a document, for example by adding noise into the input features; our work refers to what they call the *blankout corruption*, which consist of randomly deleting features in the input documents (texts or images in this case) with probability q. However, learning only from predefined adversarial examples appears sub-optimal since it is not dynamically adapted to the performance of the reader.

We think random corruption is not the most efficient way to corrupt the data, but that the corruption should be dynamically adapted to the performance of the reader. While obfuscation of one of the keywords can be too hard for the reader at the begin-

ning of the training, obfuscation of a meaningless word is unlikely to have any effect on the reader that is good enough. The learning protocol we propose aims to handle this by training jointly the obfuscation network and the reader in order to adapt the corruption difficulty to the reader's performance.

3. Adversarial reading networks

The model we propose is built to use this kind of adversariality as an adaptive dropout by challenging the reader with more and more difficult tasks during the learning. Indeed, we utilize *asymmetric self-play* to train a model called an *obfuscation network* that plays an adversarial game against a *reader*. The obfuscation network is acquiring knowledge about the reader's behaviour during the training, and it generates increasingly hard adversarial examples. Beyond increasing artificially the size of the available dataset, this adaptive behaviour of the obfuscation network prevents catastrophic forgetting phenomena of the reader. In this section, we first explain our protocol of adversarial training for robust machine comprehension and then describe the reader and obfuscation network models.

3.1. *Adversarial learning protocol*

The overall framework is a turn-based question-answering game described in Figure 2 and algorithm 1. At the beginning of each round t, the obfuscation network obfuscates one word for each document sampled from the training corpus. We fix the ratio of corrupted data / clear data to a ratio $\lambda \in [0, 1]$ of the dataset. Indeed, too low a percentage of corrupted data might not have any effect on the training and a too high one will prevent the reader of learning well. The reader is then trained on a subset of this obfuscated corpus and tested on the remaining subset. Note that both train and test sets contain corrupted data. Finally, the obfuscation network gets back a set of rewards regarding the reader performance on the obfuscated stories. Given a tuple $(d, d^\dagger, q)$ where d is the original document, $d^\dagger$ the document with an obfuscated word proposed by the obfuscation network and q the associated question, the reward r given to the obfuscation network is defined as follows:

$$r = \begin{cases} 1 & \text{if the reader answers well on } d \text{ and fail on } d^\dagger \\ 0 & \text{otherwise.} \end{cases}$$

The reward given to the obfuscation network is a direct measurement of the impact of the obfuscation on the reader performance. All the previously collected rewards are stored and used for experience replay throughout the turns. After each learning turn, all the parameters of the obfuscation network are reinitialized and retrained on all the recorded rewards. Throughout the turns, the obfuscation network accumulates information about the reader behaviour and proposes more challenging tasks as the game continues. Among the corrupted documents that the obfuscation network proposes

to the reader, 80% of the documents maximize the probability of fooling the reader from the obfuscation network point of view and 20% are randomly corrupted in order to ensure exploration. Finally, the reader keeps improving through time and any catastrophic forgetting is compensated at the next turn of the obfuscation network by focusing on these errors.

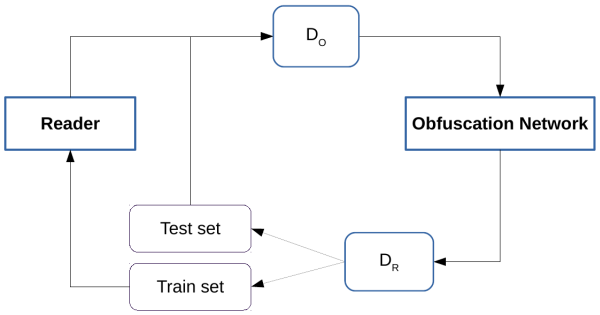

Figure 2. *Adversarial learning protocol with $D_R = \{d_i, q_i, a_i\}_i$ the reader dataset composed by tuples (document, question, answer) and $D_O = \{d_i, q_i, a_i, r_i\}_i$ the obfuscation network dataset composed by tuples (document, question, answer, reward from the reader).*

To more formally specify loss functions for the reader and the obfuscation network, let $\widehat{a}_{ij} \triangleq P(ans_{ij}|q_i, d_i^\dagger)$ denote the reader's predictive probability for ans_{ij} being the correct answer to the question q_i for $j \in [0, n]$ where n is the number of possible answers. Let us denote the index of the actual correct answer by ij^*. The reader is trained to minimize the cumulative log-loss (cross-entropy) for N questions

$$\mathscr{L}_{\text{Reader}} = -\sum_{i=1}^{N} \log \widehat{a}_{ij^*}. \qquad [1]$$

The obfuscation network is trained to fool the reader, so it suffers a loss when it fails to predict whether the reader gives a correct answer ans_{ij^*}. By denoting the indicator of the reader answering the question q_i wrong by $fail_i \in \{0, 1\}$ and obfuscation network's estimate of the probability of this failure by $\widehat{a}_i \triangleq P(fail_i = 1|q_i, d_i^\dagger)$, the obfuscation network's loss is defined as

$$\mathscr{L}_{\text{ObfNet}} = -\sum_{i=1}^{N} fail_i \log \widehat{a}_i + (1 - fail_i) \log(1 - \widehat{a}_i). \qquad [2]$$

Algorithm 1 Adversarial training

input: Let I be the initial set of data $\{(d, q, a)\}_i$ where d, q, a are sequences of index representing a document, a question and an answer.
Let A be the training set (80% of I)
Let B be the validation set (10% of I)
Let C be the testing set (10% of I)
Let D be an empty dataset
$t = 0$
while $t <$ NB_MAX_EPOCHS **do**
  Split A into A_1 (80%) and A_2 (20%)
  if t = 0 **then**
    Let $A_1^\dagger$ be A_1 with 20% of random corruption
    Let $A_2^\dagger$ be A_2 with 100% of random corruption
  else
    Reinitialize all the parameters of the obfuscation network
    Train the obfuscation network on D
    Let $A_1^\dagger$ be A_1 with 20% of data corrupted by the obfuscation network
    Let $A_2^\dagger$ be A_2 with 100% of data corrupted by the obfuscation network
  end if
  Train one epoch of the reader on $A_1^\dagger$
  for all $((d, q, a) \in A_2, (d^\dagger, q, a) \in A_2^\dagger)$ **do**
    Let r be the reward given to the obfuscation network
    if the reader succeed on d and fails on $d^\dagger$ **then**
      $D \leftarrow \{D \cup (d^\dagger, q, a, r = 1)\}$
    else if the reader succeed on d and succeed on $d^\dagger$ **then**
      $D \leftarrow \{D \cup (d^\dagger, q, a, r = 0)\}$
    end if
  end for
  Let ε_t be the empirical error of the reader on B
  if $\varepsilon_t > \varepsilon_{t-1}$ **then**
    Stop the learning
  end if
  t $\leftarrow$ t + 1
end while
Report the empirical error of the reader on C

3.2. *Baseline learning protocol*

In our reference protocol, the corruption is made by randomly obfuscating a word in several documents. This is a naive variation of the first protocol where the obfuscation network does not learn from the reader feedback at all. In fact, this protocol is similar to a dropout regularization on the embeddings layer that allows avoiding overfitting the training set. However, the obfuscation is independent of the reader per-

formance; especially, it does not take into account the difficulty of the questions. In practice, this simple adversarial protocol still improves the robustness of the results compared to a standard learning protocol. This learning protocol has strong similarities with the one proposed by Maaten *et al.* (2013).

3.3. *Reader network*

To illustrate this work, we investigate two types of neural architectures: a memory based architecture with a Gated End-to-End Memory Network (Liu and Perez, 2017b) (`GMemN2N`) and a multi-layer attention based architecture largely inspired by the recent `R-Net`(Wang *et al.*, 2017b) excepted for its output layer, adapted to the format of the datasets used in this work. These two architectures are state-of-the-art models for machine reading and most of the recent models are a combination of layers included in these two architectures. Paragraphs below describe these two architectures and how we have integrated them in the adversarial learning protocol.

3.3.1. *Gated End-to-End Memory Network reader*

The first model used as a reader is a Gated End-to-End Memory Network (Liu and Perez, 2017b), `GMemN2N` (Figure 3). This architecture is based on two different memory cells and an output prediction. An input memory representation $\{m_i\}$ and an output representation $\{c_i\}$ are used to store embedding representations of inputs. Suppose that an input of the model is a tuple (d, q) where d is a document, i.e., a set of sentences $\{s_i\}$, and q is a query about d. The entire set of sentences is converted into input memory vectors $m_i = A\Phi(s_i)$ and output memory vectors $c_i = C\Phi(s_i)$ by using two embedding matrices A and C. The question q is also embedded using a third matrix B, $u = B\Psi(q)$ of the same dimension as A and C, where Φ and Ψ are respectively the document embedding function and the question embedding function described in the next paragraph. The input memory is used to compute the relevance of each sentence in its context regarding the question, by computing the inner product of the input memory sentence representation with the query. A softmax is then used to map the inner product to a probability. The response $o = \sum_i p_i c_i$ from the output memory is the sum of the output memory vectors $\{c_i\}$ weighted with the sentence relevancies calculated before $p_i = \text{softmax}(\boldsymbol{u}^T m_i)$. A gated mechanism is used when we update the value of the controller u:

$$T^k(u^k) = \sigma(W_T^k u^k + b_T^k), \quad [3]$$

$$\mathbf{u^{k+1}} = o^k \odot T^k(u^k) + u^k \odot (1 - T^k(u^k)), \quad [4]$$

where W_T^k are matrices of size $d \times d$ and b_T^k a vector of size d with d the size of the memory cells.

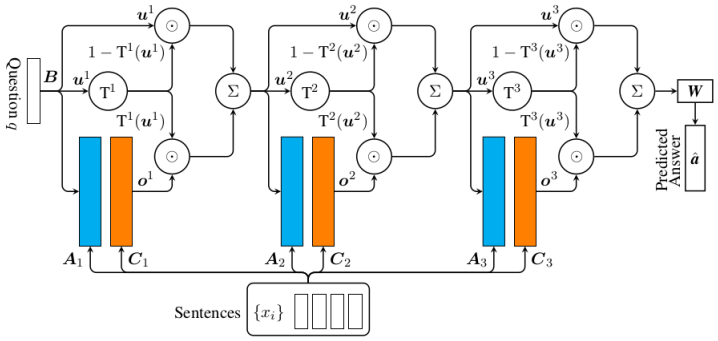

Figure 3. *Gated End-to-End Memory Network (Liu and Perez, 2017b).*

Assuming we use a model with K hops of memory, the final prediction is:

$$\widehat{a} = \text{softmax}(W(o^K + u^K) + b), \quad [5]$$

where W is a matrix of size $d \times v$ and b a vector of size d with v the number of candidate answers. In this model, we do not use the adjacent or layer-wise weight tying scheme and all the matrix A^k and B^k of the multiple hops are different.

Text and question representations (Figure 4): To build the sentence representations, we use a 1-dimensional Convolutional Neural Network (CNN) with a list of filter sizes over all the sentences as proposed in Kim (2014). Let $[s_1, \ldots, s_N]$ be the vectorial representation of a document with N sentences where $s_i = [w_{i,1}, w_{i,2}, \ldots, w_{i,n}]$ is the $i - th$ sentence which contains n words. Given a convolutional filter $F \in \mathbb{R}^{h \times d}$ where h is the width of the convolutional window, i.e, the number words it overlaps, the convolutional layer produces:

$$c_{i,j} = f(F \odot [Ew_{i,j}, \ldots, Ew_{i,j+h}]), \forall j \in [1, n - j], \quad [6]$$

where $\odot$ is the element-wise multiplication, f a rectified linear unit (ReLU) and E is the embedding matrix of size $d \times V$ where V is the vocabulary size and d the word embedding size. Then, a max pooling operator is applied to this vector to extract features. Given a filter F, after a convolutional operation and a max pooling operation, we obtain a feature $\hat{c}_i = \max_j(c_{i,j})$ from the i^{th} sentence of the text. Multiple filters with varying sizes are used. Assume that our model uses N_s different filter sizes and N_f for each size, we are able to extract $N_s \times N_f$ features for one sentence. The final representation of the sentence is the concatenation of the extracted features from all the filters:

$$\Phi(s_i) = [\hat{c}_{iF_1}, \hat{c}_{iF_2}, \ldots, \hat{c}_{iF_{N_s * N_f}}]. \quad [7]$$

Compared to an LSTM encoding the CNN layer is faster and gives better results on the different tasks we evaluated our model. This result seems coherent with recent

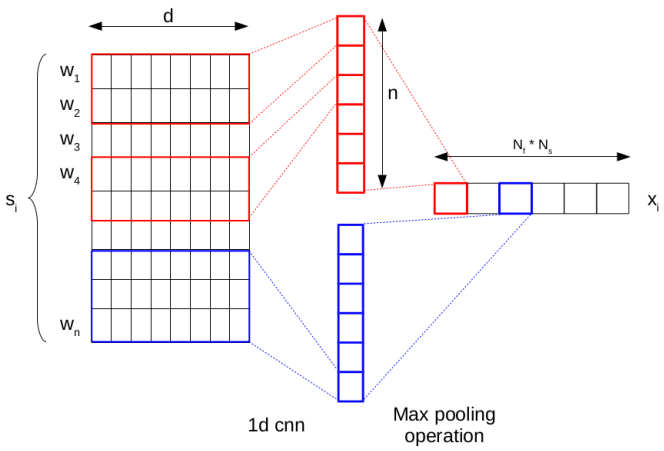

Figure 4. *An encoded sentence where d is the word embedding size, N_f the number of filters of each size and N_s the number of different filter sizes used.*

results of Dauphin *et al.* (2017). We use a bidirectional GRU to encode the question. The question representation $\Psi(q)$ is the concatenation of the final states of the forward and backward GRU on this question.

3.3.2. *R-Net based network*

The second architecture investigated in this article is based on the state-of-the-art `R-Net` model (Wang *et al.*, 2017b). The main part of the architecture remains the same as the original model, except for the last layer. We replaced the pointer network, originally used to select in the document the span of text that corresponds to the answer, by a fully connected layer followed by a softmax to output the probability of each candidate word to be the answer. The following lines describe the structure of this architecture, composed of multiple stacked layers.

Encoding layer: Each sentence is tokenized by word and each token is represented by the concatenation of the word level, and character level embeddings. The word level embedding is computed via a lookup table initialized with `GloVe` pre-trained embeddings and the character embedding of a token is the final state of a GRU network over the sequence of its characters. Finally, these tokens are fed to a Recurrent Neural Network (RNN) and the document and question are represented by the intermediate states of this RNN.

Gated question/document attention: Assuming that $d = \{u_i^d\}_{i=0}^N$ and $q = \{u_i^q\}_{i=0}^N$ are the sequences of embedding tokens of the document and the question after the encoding layer with N the length of the document and n the length of the

question. Then we compute an attention between the representation of the question and each token of the document. The document is transformed to $d = \{v_i^d\}_{i=0}^N$ with:

$$v_i^d = RNN(v_{i-1}^d, [u_i^d, c_i]),$$

where c_i is an attention vector of the question over the token i of the document. This layer produces a question-aware representation of the document.

Self-attention: So far each token contains information from the question due to the question/document attention layer and from its surrounding context due to the RNN at the end of the encoding part but does not handle long-term dependencies inside the document. The self-attention layer produces an attention between the whole document and each individual token of it. $d = \{h_i^d\}_{i=0}^N$ with:

$$h_i^d = BiRNN(h_{i-1}^d, [v_i^d, c_i]),$$

where c_i is an attention vector of the whole document over the token i.

Output layer: The decision support is the concatenation of the h_i, for $i \in [0, N]$. $o^d = \text{concat}(\{h_i\}_{i=0}^N)$ and finally:

$$\widehat{a} = \text{softmax}(Wo^d + b),$$

where W is a matrix of size $N * d \times v$ and b a vector of size d with v the number of candidate answers.

3.4. *Obfuscation network*

The objective of this model is to predict the probability of the reader to successfully respond to a question about a document with an obfuscated word. This estimate will be used by the obfuscation network to determine the position of the obfuscated word in the document which maximizes the probability of the reader to fail its task. We use a similar architecture as the reader, i.e a `GMemN2N` when the reader is a `GMemN2N` and a `R-Net` when the reader is a `R-Net`. However, on the last layer, a sigmoid function is used to predict the probability of the reader to fail on this input: Assuming that o is the decision support of the obfuscation network, then:

$$\widehat{a} = \sigma(Wo + b), \quad [8]$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ and $\widehat{a} \in [0, 1]$ is the predicted probability of failure of the reader and W a matrix of size $d \times 1$. We impose this symmetry between the architecture of the reader and of the obfuscation network in order to keep a fair challenge between the two adversary networks.

An input to the reader is a tuple $(d^\dagger, q)$ where $d^\dagger$ is a document with an obfuscated word. To obfuscate a word, we replace it by the word *unk* for *unknown*. The output of the obfuscation network is a real number $r \in [0, 1]$ which is the expected probability of the reader to fail on the question. The objective of the obfuscation network is

to select the corrupted document which maximizes this reward. We use the same text passage and query representation as for the reader, based on a CNN with different filter sizes for the document and the two last hidden states of a bidirectional Gated Rectified Unit (GRU) recurrent network for the question encoding for the `GMemN2N` and based on character and word level embeddings for the `R-Net`. Both models are fully differentiable.

4. Datasets and data preprocessing

Cambridge Dialogs: The transactional dialog corpus proposed by Rojas-Barahona *et al.* (2017) has been produced by a crowdsourced version of the Wizard-of-Oz paradigm. It was originally designed for dialog state tracking, but Liu and Perez (2017a) have shown that this task could also be considered as a reading task. In such setting, the informable slots provided as metadata to each dialog were used to produce questions for a dialog comprehension task. The dataset deals with an agent assisting a user to find a restaurant in Cambridge, UK. To propose the best matching restaurant, the system needs to extract 3 constraints which correspond to the informable slots in the dialog state tracking task: *Food, Price range, Area*. Given a dialog between an agent and a user, these informable slots become questions for the model we propose. The dataset contains 680 different dialogs about 99 different restaurants. We preprocess the dataset to transform it into a question-answering dataset by using the three informable slot types as questions about a given dialog. After this preprocessing operation, we end up with our question-answering formatted dataset which contains 1,352 possible answers.

Document: *I want the phone number of a moderately priced restaurant with Spanish food.* *La Tasca would fit the bill. Its phone number is 01223 464630.* *Can you tell me what area of town it is located?* *La Tasca is located in the center part of town.* *Thank you, goodbye.* *You're welcome.* **Question:** *What is the area?* **Answer:** *Center.*

Table 1. *An example from the Cambridge dataset formatted for question-answering task.*

TripAdvisor aspect-based sentiment analysis: This dataset contains a total of 235K detailed reviews extracted from the TripAdvisor website and originally released by Wang *et al.* (2010). These reviews represent around 1,850 hotels. Each review is associated to an overall rating, between 0 and 5 stars. Furthermore, 7 aspects: *value, room, location, cleanliness, checkin/front desk, service,* and *business service* are available. We transform the dataset into a question-answering task over a given review. Concretely, for each review, a question is an aspect and we use the number of

stars as the answer. This kind of machine-reading approach to sentiment analysis was previously proposed in Tang *et al.* (2016).

Document: *Service was ok, staff helpful, room was basic, marks on bedding top cover looked like blood, sheets clean, bathroom not so nice, broken tiles on floor, shower head was disgusting and needed to be replaced, location was good, close to the metro and the Colosseum, both only a 10 min walk, liked that the hotel was close to many cafe's restaurant's, disliked the shower in room.*

Question: *How is the cleanliness?*
Answer: 2/5.

Question: *How is the service?*
Answer: 3/5.

Table 2. *An example from the TripAdvisor dataset.*

Children's Book Test (CBT): The dataset is built from freely available books (Hill *et al.*, 2015) produced by Project Gutenberg [1]. The training data consists of tuples (S, q, C, a) where S is the *context* composed of 20 consecutive sentences from the book, q is the *query*, C a set of 10 *candidate answers* and a the *answer*. The query q is the 21^{st} sentence, i.e., the sentence that directly follows the 20 sentences of the *context* and where one word is removed and replaced by a missing word symbol. Questions are grouped into 4 distinct categories depending of the type of the removed word: Named Entities (NE), (Common) Nouns (CN), Verbs (V) and Prepositions (P). This division of answers according to the type of the word that has been removed give a way to evaluate the performance of a model in different situations. It provides relevant information on the strengths and weaknesses of a given architecture. The training contains 669,343 inputs (context+query) and we evaluated our models on the provided test set which contains 10,000 inputs, 2,500 per category. This dataset evaluates the capability that a model has to predict a word based on its context.

1. `https://www.gutenberg.org`.

Document:
1 When she got home she shut herself up in her room and cried.
2 There was nothing for her to do but resign, she thought dismally.
3 On the following Saturday Esther went for an afternoon walk, carrying her Kodak with her.
4 It was a brilliantly fine autumn day, and woods and fields were basking in a mellow haze.
19 Bob and Alf Cropper were up among the boughs picking the plums.
20 On the ground beneath them stood their father with a basket of fruit in his hand.

Question: *21 Mr. Cropper looked at the **XXXXX** and from it to Esther.*
Answer: *proof*
Candidates: *Saturday* | *boughs* | *face* | *father* | *home* | *nothing* | *proof* | *remarks* | *smile* | *woods*

Table 3. *An example from the CBT dataset.*

5. Experiments

In this section, we present our experimental settings and the results of this adversarial training protocol on the three datasets presented in Section 4.

5.1. *Training details*

10% of the dataset was randomly held-out to create a test set. We split the dataset before all the training operations and each protocol was tested on the same test dataset. For the training phase, we split the training dataset to extract a validation set to perform early stopping. We used Adam optimizer (Kingma and Ba, 2014) with a starting learning rate of 0.0005. We set the dropout to 0.9 which means that during training, randomly selected 10% of the parameters are not used during the forward pass and not updated during the backward propagation of error. We also added the gated memory mechanism (Liu and Perez, 2017b) that dynamically regulates the access to the memory blocks. This mechanism had a very positive effect on the overall performance of our models. All weights were initialized randomly from a Gaussian distribution with zero mean and a standard deviation of 0.1. We augmented the loss with the sum of squares of the model parameters.

The hyperparameters have been chosen via cross-validation on the validation set of the different datasets. We set the batch size to 16 inputs and we used word embeddings of size 300. We initialized all the embedding matrices with pre-trained `GloVe` word vectors (Pennington *et al.*, 2014) and used random vectors for the words not present in

the GloVe. It seems that for our experiments CNN encoding does not improve only the overall accuracy of the model compared to LSTM but also the stability by decreasing the variance of the results. So, in practice, we used 128 filters of size 2, 3, 5 and 8 resulting in a total of 512 filters for the one-dimensional convolutional layer.

We repeated each training 10 times for the first two datasets and report maximum and average accuracy. The average value corresponds to the average score over the 10 runs on the test set. Maximum value corresponds to the score on the test set achieved by the model that performed best on the validation set. During the adversarial learning, the dataset contained 70% of clear dialogs and 30% of corrupted dialogs, $\lambda = 0.3$. Inside these corrupted data, 20% were randomly obfuscated by the obfuscation network in order to make it learn from exploration and the obfuscation network maximized its reward for the remaining 80%. Due to the format of the dataset, we slightly modified the output layer of our reader for the CBT task. Instead of projecting on a set of candidate answers, the last layer of the reader made a projection on the entire vocabulary $\widehat{\boldsymbol{a}} = \sigma(M \odot W(o^K + u^K))$ where W is a matrix of size $V \times d$ with V the vocabulary size, $\odot$ the elementwise product and M the mask vector of size V containing 1 if the corresponding word is proposed in the candidate answers, and 0 otherwise.

5.2. *Results*

In this section, we report the results of our implementation of two baselines: a simple logistic regression and an Attention-Sum Reader (Kadlec *et al.*, 2016). Then we present the results of our implementation of the two neural architectures presented in Section 3.3, trained with the standard training, the *uniform* training, which is the reader trained with the baseline protocol 3.2 and with our adversarial learning protocol 3.1.

	Log Reg	ASR	GMemN2N			uniform GMemN2N			adversarial GMemN2N		
hops			4	5	6	4	5	6	4	5	6
Max	58.4	40.8	82.1	85.8	80.6	85.1	85.8	82.8	82.8	79.8	**88.1**
Mean	58.2	39.5	76.9	74.8	74.2	77.4	77.7	74.9	**79.8**	77.8	79.6

	R-Net	uniform R-Net	adversarial R-Net
Max	88.1	89.5	**90.8**
Mean	87.5	89.2	**90.0**

Table 4. *Average and maximum accuracy (%) on the Cambridge dataset on 10 replications. In bold, the best result per architecture.*

	Log Reg	ASR	GMemN2N			uniform GMemN2N			adversarial GMemN2N		
hops			4	5	6	4	5	6	4	5	6
Max	59.4	45.2	62.3	62.4	60.5	63.1	61.4	63.1	**64.6**	63.5	62.3
Mean	59.0	42.3	60.8	60.6	58.5	62.3	60.3	59.6	**62.8**	61.2	60.8

	R-Net	uniform R-Net	adversarial R-Net
Max	62.3	63.8	**64.5**
Mean	61.9	62.2	**63.0**

Table 5. *Average and maximum accuracy (%) on the TripAdvisor dataset on 10 replications. In bold, the best result per architecture.*

Tables 4 and 5 display the scores obtained by these models on the Cambrige and TripAdvisor datasets. Each experiment was run 10 times and we report in this table the maximum score on the test set (based on the validation set) and the average score. The precise number of hops needed to achieve the best performance with the GMemN2N is not obvious, so we present all the results for readers and obfuscation networks between 4 and 6 hops.

We observe that the **adversarial learning protocol improves the accuracy** of the GMemN2N and R-Net compared to the standard and uniform training protocol for all the experiments.

We improve the score of the reader by 2.3 points on the Cambridge task for a GMemN2N with 6 hops compared to the standard training. This adversarial protocol, applied to the R-Net architecture, improves the average score by 2.5 points on this dataset.

The best performance on the TripAdvisor dataset was achieved by the adversarial R-Net. On 10 replications of the experiment, the average accuracy of this model was improved by 1.1 points compared to the standard approach.

The GMemN2N with 4 hops achieved the best performance of this architecture. The accuracy was improved by 1.5 points when the model was trained with our adversarial protocol.

The uniform protocol improves the stability of the performance compared to a standard reader but further improvements were obtained with the adversarial protocol which improved both the overall accuracy and the stability of the performance. Indeed the variance of the results decreased when the training was done with the adversarial protocol, especially for the GMemN2N. Such architecture does not always converge to the optimal minima and the adversarial learning, acting as an adaptive dropout, seems to help the model to generalized better. It is not clear, for this task, whether the number of hops, between 4 and 6, affects the general behaviour, but we achieved the best performance with our adversarial protocol and a reader with 6 hops.

	Log Reg				ASR			
Task	P	V	NE	CN	P	V	NE	CN
Max	56.3	37.1	26.5	25.6	24.7	32.7	22.1	18.3

	GMemN2N				uniform GMemN2N				adversarial GMemN2N			
Task	P	V	NE	CN	P	V	NE	CN	P	V	NE	CN
Max	56.0	58.5	31.9	39.0	58.1	53.6	31.6	34.0	**71.1**	**60.4**	**35.3**	**39.4**

	R-Net				uniform R-Net				adversarial R-Net			
Task	P	V	NE	CN	P	V	NE	CN	P	V	NE	CN
Max	55.0	68.3	44.0	42.6	56.3	68.9	43.8	40.7	**60.0**	**70.0**	**44.5**	**42.9**

Table 6. *Accuracy (%) on the CBT dataset. In bold, the best result per architecture.*

Performance on the CBT dataset are displayed in Table 6. Because of the size of this dataset, we didn't repeat the training 10 times but only once. Results of the uniform training seem similar to the performance of the standard reader in this case but **the accuracy of the models trained with our adversarial protocol remains higher than others'**. This last experiment shows that augmenting the data distribution with random adversarial examples might not help the training as it was explained by Jia and Liang (2017). But we show that even in that case the protocol we propose is able to generate smart adversarial examples that will finally help the reader to improve its overall performance.

6. Visualizations and analysis

In this section, we present a series of analysis of the behavior of the competitive networks to better understand how the adversarial setting affects the training. We propose to analyze the probabilities of obfuscation of the different words of a given input (d, q, a).

In order to better understand how the obfuscation network learns from the reader behaviour during the adversarial protocol, Figure 5 depicts the rewards that the obfuscation network expects for each word of a document after several rounds of the game. Given a tuple (d, q) where d is a clear document and q a query, and assuming the document contains k words, we generate k corrupted documents where one word is obfuscated in each of them. We then feed the obfuscation network with these corrupted data and report the results. The expected rewards from the reader are displayed in green on the document. A strong intensity means that a high reward is expected.

Document:
A cheap Jamaican restaurant, please.
If you can not find one, then Mediterranean.
There are no Jamaican restaurants listed. I will look for Mediterranean.
Okay what is the phone number and postcode of that Mediterranean place?
The Gardenia is located in the centre at postcode C.B 2, 3 L.L and can be reached at 01223 356354. Can I get anything else for you?
Thank you.
Thank you for using the Cambridge restaurant system. Goodbye.

Question: What is the pricerange?

Document:
I am looking for a restaurant in the center of town that serves European food.
Eraina would fit the bill.
What are the address, phone number, and price range?
It's pretty expensive. It's at Free School Lane City Centre, and you can call them at 01223 368786. Anything else I can help you with?
No, that will be all. Thanks!
You're welcome!

Question: What is the area?

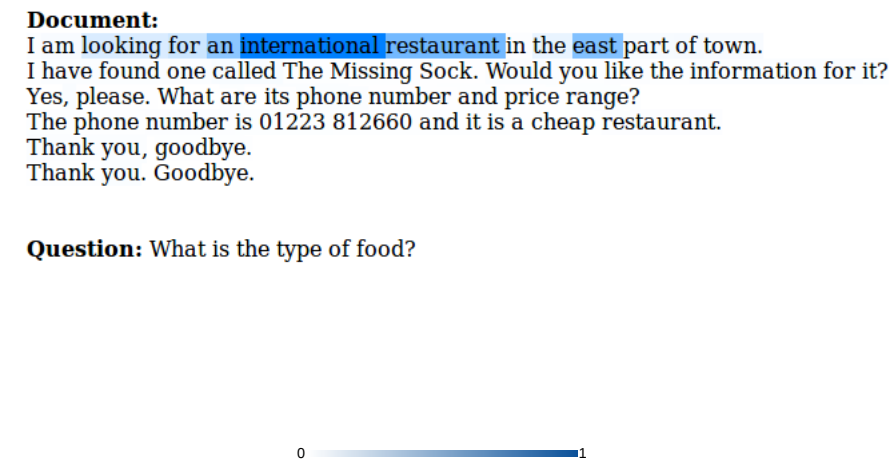

Figure 5. *Rewards expected by the obfuscation network after 100 rounds over a Cambridge dialog.*

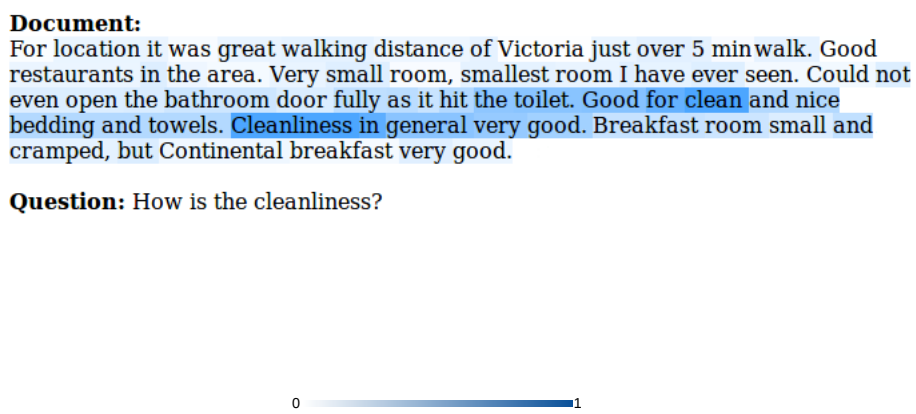

Figure 6. *Rewards expected by the obfuscation network after 100 rounds over a TripAdvisor review.*

We see that the obfuscation network tends to obfuscate some important keywords of the dialogs in Figure 5. Furthermore, the obfuscation network is not pointing on a single word but it points on a word and on its neighborhood. This could be a consequence of the encoding which is not only a representation of a single word but a representation of a word in its context. In Figure 6, we can see that the obfuscation network tends to affect a high probability of getting a reward for multiple words of the review. This can be a consequence of the performance of the reader on this dataset. Indeed if the reader is not generally confident about its answers, small changes in the reviews could lead to fool it. However, we can see on the figure that the most probable regions obfuscated by the obfuscation network refer to the cleanliness of the hotel which is coherent with the question.

7. Conclusion and future work

In this paper, we propose an adversarial learning protocol to train coupled deep neural networks for the task of machine reading. We propose two baselines, a Logistic Regression and an Attention-Sum Reader, on the three datasets used for our experiments. Then we experiment our adversarial learning protocol on two main types of neural architectures based on state-of-the-art machine reading models: a `GMemN2N` and a `R-Net`. In addition, we compare our adversarial protocol to a protocol based on a uniform corruption of data.

On all the reported experiments, the models trained with our novel protocol outperform the equivalent models trained with a standard supervised protocol or a protocol that introduces a uniform noise in the data which correspond to the more classic approach of dropout. Moreover, our adversarial protocol seems to improve the stability of the models' performance. Indeed, the variance of the results decreased when the training was done in an adversarial setup. We propose several visualizations that allow interpreting how the reader produces an answer, and which parts of the document are crucial for it to take its decision.

In future work, we plan to improve this novel protocol through an active question-answering task. Indeed the choice to only let the obfuscation network remove a single word might not be optimal. We would like to let it obfuscates multiple words while letting the reader the possibility to ask for revealing several words that might help it during training. Finally, we are currently investigating an adaptation of this protocol to Visual Question Answering.

The lack of robustness against adversarial examples and the difficulty to train deep neural networks with a limited set of data is not a specificity of language processing. This adversarial way of training deep neural networks should not be restricted to text documents but we think that it can also be useful in other domains.

8. References

Chen D., Bolton J., Manning C. D., "A Thorough Examination of the CNN/Daily Mail Reading Comprehension Task", *CoRR*, 2016.

Cui Y., Chen Z., Wei S., Wang S., Liu T., Hu G., "Attention-over-Attention Neural Networks for Reading Comprehension", *ACL*, 2017.

Dauphin Y., Fan A., Auli M., Grangier D., "Language Modeling with Gated Convolutional Networks", *ICML*, 2017.

Goodfellow I. J., Pouget-Abadie J., Mirza M., Xu B., Warde-Farley D., Ozair S., Courville A. C., Bengio Y., "Generative Adversarial Nets", *NIPS*, 2014a.

Goodfellow I. J., Shlens J., Szegedy C., "Explaining and Harnessing Adversarial Examples", *CoRR*, 2014b.

Guo X., Klinger T., Rosenbaum C., Bigus J. P., Campbell M., Kawas B., Talamadupula K., Tesauro G., "Learning to Query, Reason, and Answer Questions On Ambiguous Texts", 2017.

Hermann K. M., Kociský T., Grefenstette E., Espeholt L., Kay W., Suleyman M., Blunsom P., "Teaching Machines to Read and Comprehend", *CoRR*, 2015.

Hill F., Bordes A., Chopra S., Weston J., "The Goldilocks Principle: Reading Children's Books with Explicit Memory Representations", *CoRR*, 2015.

Jia R., Liang P., "Adversarial Examples for Evaluating Reading Comprehension Systems", *Empirical Methods in Natural Language Processing (EMNLP)*, 2017.

Kadlec R., Schmid M., Bajgar O., Kleindienst J., "Text Understanding with the Attention-Sum Reader Network", *CoRR*, 2016.

Kim Y., "Convolutional Neural Networks for Sentence Classification", *EMNLP*, 2014.

Kingma D. P., Ba J., "Adam: A Method for Stochastic Optimization", *CoRR*, 2014.

Liu F., Perez J., "Dialog state tracking, a machine reading approach using Memory Network", *EACL*, 2017a.

Liu F., Perez J., "Gated End-to-End Memory Networks", *EACL*, 2017b.

Maaten L., Chen M., Tyree S., Weinberger K. Q., "Learning with Marginalized Corrupted Features", *in* S. Dasgupta, D. Mcallester (eds), *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, vol. 28, JMLR Workshop and Conference Proceedings, p. 410-418, 2013.

Miyato T., Dai A. M., Goodfellow I. J., "Virtual Adversarial Training for Semi-Supervised Text Classification", *CoRR*, 2016.

Nguyen T., Rosenberg M., Song X., Gao J., Tiwary S., Majumder R., Deng L., "MS MARCO: A Human Generated MAchine Reading COmprehension Dataset", *CoRR*, 2016.

Pennington J., Socher R., Manning C. D., "GloVe: Global Vectors for Word Representation", *Empirical Methods in Natural Language Processing (EMNLP)*, p. 1532-1543, 2014.

Rajpurkar P., Zhang J., Lopyrev K., Liang P., "SQuAD: 100, 000+ Questions for Machine Comprehension of Text", *EMNLP*, 2016.

Richardson M., Burges C. J., Renshaw E., "MCTest: A Challenge Dataset for the Open-Domain Machine Comprehension of Text", *Proc. EMNLP*, 2013.

Rojas-Barahona L. M., Gasic M., Mrksic N., Su P.-H., Ultes S., Wen T.-H., Young S. J., Vandyke D., "A Network-based End-to-End Trainable Task-oriented Dialogue System", *EACL*, 2017.

Seo M. J., Kembhavi A., Farhadi A., Hajishirzi H., "Bidirectional Attention Flow for Machine Comprehension", *CoRR*, 2016.

Shen Y., Huang P.-S., Gao J., Chen W., "ReasoNet: Learning to Stop Reading in Machine Comprehension", *CoCo@NIPS*, 2016.

Silver D., Huang A., Maddison C. J., Guez A., Sifre L., van den Driessche G., Schrittwieser J., Antonoglou I., Panneershelvam V., Lanctot M., Dieleman S., Grewe D., Nham J., Kalchbrenner N., Sutskever I., Lillicrap T. P., Leach M., Kavukcuoglu K., Graepel T., Hassabis D., "Mastering the game of Go with deep neural networks and tree search", *Nature*, vol. 529 7587, p. 484-9, 2016.

Sordoni A., Bachman P., Bengio Y., "Iterative Alternating Neural Attention for Machine Reading", *CoRR*, 2016.

Srivastava N., Hinton G. E., Krizhevsky A., Sutskever I., Salakhutdinov R., "Dropout: a simple way to prevent neural networks from overfitting", *Journal of Machine Learning Research*, vol. 15, nº 1, p. 1929-1958, 2014.

Tang D., Qin B., Liu T., "Aspect Level Sentiment Classification with Deep Memory Network", *EMNLP*, 2016.

Tesauro G., "Temporal Difference Learning and TD-Gammon", *Commun. ACM*, vol. 38, nº 3, p. 58-68, March, 1995.

Trischler A., Wang T., Yuan X., Harris J., Sordoni A., Bachman P., Suleman K., "NewsQA: A Machine Comprehension Dataset", *Rep4NLP@ACL*, 2017.

Trischler A., Ye Z., Yuan X., Bachman P., Sordoni A., Suleman K., "Natural Language Comprehension with the EpiReader", *EMNLP*, 2016.

Wang H., Lu Y., Zhai C., "Latent Aspect Rating Analysis on Review Text Data: A Rating Regression Approach", *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, ACM, New York, NY, USA, p. 783-792, 2010.

Wang T., Yuan X., Trischler A., "A Joint Model for Question Answering and Question Generation", *CoRR*, 2017a.

Wang W., Yang N., Wei F., Chang B., Zhou M., "Gated Self-Matching Networks for Reading Comprehension and Question Answering", *ACL*, 2017b.

Weston J., Bordes A., Chopra S., Mikolov T., "Towards AI-Complete Question Answering: A Set of Prerequisite Toy Tasks", *CoRR*, 2015.

Weston J., Chopra S., Bordes A., "Memory Networks", *CoRR*, 2014.

Yu H., Munkhdalai T., "Neural Semantic Encoders", *Proceedings of the conference. Association for Computational Linguistics. Meeting*, vol. 1, p. 397-407, 2017.

Yuan X., Wang T., Gülçehre C., Sordoni A., Bachman P., Zhang S., Subramanian S., Trischler A., "Machine Comprehension by Text-to-Text Neural Question Generation", *Rep4NLP@ACL*, 2017.