Dialogue system for QA forum websites February 2019

Universidad del País Vasco — Euskal Herriko Unibertsitatea

Building a dialogue system for question-answer forum websites

Author: Jon Ander Campos

Tutors: Eneko Agirre and Arantxa Otegi

Assessors: Aitor Soroa

hap/lap

Hizkuntzaren Azterketa eta Prozesamendua
Language Analysis and Processing

Final Thesis

February 2019

Departments: Computer Systems and Languages, Computational Architectures and Technologies, Computational Science and Artificial Intelligence, Basque Language and Communication, Communications Engineer.

Laburpena

Dialogo-sistemak gizakiak laguntzeko sistema automatikoak dira, eta beren ezaugarri nagusia da komunikazioa hizkuntza naturalaren bidez gauzatzeko gai direla. Azken boladan bultzada handia jaso eta eguneroko tresnetan aurkitu daitezke (*Siri*, *Cortana*, *Alexa*, etab.). Sistema hauen erabilera handitu ahala, *Community Question Answering* (CQA) edo *Frequently Asked Questions* (FAQ) direlakoak dialogo bitartez atzitzeko interesa zeharo handitu da, bereziki enpresa munduan. Egungo dialogo sistemen elkarrizketarako ahalmena, ordea, oso mugatua da, eskuzko erregelen bidez definituta baitaude. Horrek domeinu berri batean ezartzeko edo behin produkzioan martxan dagoenean monitorizatu eta egokitzeko kostuak handitzen ditu. Bestalde, nahiz eta ikaskuntza sakona bezalako teknikek oso emaitza onak lortu dituzten Hizkuntzaren Prozesamenduko alor desberdinetan, asko sufritzen dute datu eskasiaren arazoa, datu kopuru izugarriak behar baitituzte ikasketarako. Hemen aurkeztutako proiektuaren helburu nagusia aipatutako mugak arintzea da, sare neuronaletan oinarritutako sistema bat inplementatuz eta sistema hauen etorkizuneko garapena bultzatu eta errazteko CQA datu multzo bat sortuz.

Abstract

Dialogue-systems are automatic systems developed for helping humans in their daily routines. The main characteristic of these systems is that they are able to communicate using natural language. Lately, dialogue agents are becoming increasingly trendy and are already part of our lives as they are implemented in many tools (*Siri*, *Cortana*, *Alexa*...). This incursion of voice agents has increased the interest of accessing Community Question Answering (CQA) and Frequently Asked Questions (FAQ) information by dialogue means, specially in the industrial world. Nowadays, dialogue systems have their conversational ability very limited as they are defined by hand-crafted rules. This hand-crafted nature, makes domain adaptation an extremely costly and time consuming task. On the other hand, deep learning based techniques, that have achieved state-of-the-art results in many Natural Language Processing (NLP) tasks, suffer from lack of data as they need huge amounts of labelled records for training. So, the main aim of this project, is to develop a neural system together with a CQA dataset for enabling future research in CQA dialogue systems.

Acknowledgements
This work has been partially funded by the Spanish Research Agency LIHLITH project (PCIN-2017-118 / AEI) in the framework of EU ERA-Net CHIST-ERA.

Contents

1 Introduction **1**
1.1 Motivation and goals . . . 2

2 Background **3**
2.1 Neural Methods for NLP . . . 3
2.2 Neural Question Answering . . . 7
2.2.1 Methods for Question Answering . . . 8
2.2.2 Neural Question Answering Systems . . . 11
2.2.3 Question Answering Datasets . . . 15
2.3 Neural Dialogue Systems . . . 17
2.3.1 Methods for Dialogue . . . 18
2.3.2 Neural Dialogue Systems . . . 20
2.3.3 Dialogue Datasets . . . 21

3 Developed System **26**
3.1 BERT original system . . . 26
3.1.1 Architecture . . . 26
3.1.2 Input . . . 28
3.1.3 Pre-training . . . 29
3.1.4 Fine-tuning for QA . . . 31
3.1.5 Available pre-trained models . . . 33
3.2 BERT fine tuning for dialogue . . . 34
3.3 Experiments . . . 35

4 Developed Dataset **38**
4.1 Cooking thread selection . . . 39
4.2 Dataset collection . . . 40
4.2.1 Interactive Task . . . 40
4.3 Trials and tribulations with AMT . . . 42
4.4 Data collection and analysis . . . 44

5 Conclusion and future work **47**

A Collected Dialogues **53**

List of Figures

1 An example of a feed forward neural network. In this network the information from the input layer is moved and transformed through the hidden layer to the output layer. Source: Wikipedia, `https://en.wikipedia.org/wiki/Feedforward_neural_network` 4

2 An example of a RNN. From bottom to top: input state (x), hidden state (h) and output state (o). U, W and V are the weights of the network. The unfolded version of the RNN can be seen in the right side. Source: Wikipedia, `https://en.wikipedia.org/wiki/Recurrent_neural_network` 5

3 An example of a CNN where given an input image two convolution and subsampling layers are used. Source: Wikipedia, `https://en.wikipedia.org/wiki/Convolutional_neural_network` 5

4 Visual representation of C-BoW and Skip-gram. Given context words $\{w(t-2), w(t-1), w(t+1), w(t+2)\}$ C-BoW sums their representations and tries to predict the target word $w(t)$. On the contrary, Skip-gram tries to predict context words $\{w(t-2), w(t-1), w(t+1), w(t+2)$ given $w(t)\}$. Source: (Mikolov et al., 2013) 6

5 An example from the SQUAD 1.1 dataset where a factoid question is shown with the ground truth answer spotted in the text. (Rajpurkar et al., 2016) 8

6 A training example taken from SQUAD 2.0 dataset (Rajpurkar et al., 2018). Here, the passage p, the question q and the answer a, which is spotted in the text, can be appreciated. 10

7 Architecture of the BIDAF system. In the left side of the picture the six main layers of the system can be appreciated. Source: (Seo et al., 2016) . . 12

8 An example of a character based CNN for mapping the word "Iparragirre" to high-dimensional vector space. 13

9 Architecture of the extended BIDAF++ system. Source: (Clark and Gardner, 2017) 15

10 Standard interpretation of the Turing test (Turing, 1950) 18

11 A conversation about invented baseball players. Source : (Jurafsky and Martin, 2014) 19

12 Some examples of dialogue acts. Source : (Jurafsky and Martin, 2014) . . . 19

13 Architecture of a dialogue-state system. Source : (Jurafsky and Martin, 2014) 20

14 The questioner interface from CoQA. Source : (Reddy et al., 2018) 23

15 The answerer interface from CoQA. Source : (Reddy et al., 2018) 24

16 Visual representation of the interactive task proposed in QuAC. Source: poster of the QuAC dataset. 25

17 Example of the collected dialogues in QuAC where the arrows define the continuation set of dialogue acts. Source: (Choi et al., 2018). 25

18 Architecture of the encoder of the Transformer. Source: (Vaswani et al., 2017). 26

19 In the left the Scaled Dot-Product Attention can be appreciated. In the right the Multi-Head Attention that consists of several attention layers in parallel. Source: (Vaswani et al., 2017). 28
20 Input representation of the BERT model. The input embeddings are the sum of the token, segmentation and positional embeddings. Source: (Devlin et al., 2018). 29
21 Visual representation of the Masked LM pre-training task. 31
22 Visual representation of the Next-Sentence Prediction pre-training task. . . 32
23 SQuAD fine-tuning approach of the BERT model. Source: (Devlin et al., 2018) 33
24 $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ models with 12 and 24 stacked encoders respectively. 34
25 Modified BERT fine-tuning approach for dialogue act prediction. Source of original: (Devlin et al., 2018) 35
26 Example of a thread from the Stackexchange cooking community. 39
27 Interface of the curious role from the proposed dialogue collection interactive task. 41
28 Interface of the cooking expert role from the proposed dialogue collection interactive task. 42
29 An example of the dialogues obtained in the second AMT trial of our interactive task for mining cooking dialogues. 44

List of Tables

1 Some examples from the SPIDER dataset (Yu et al., 2018b). In the left column the text question is given and in the right column the SQL language mapping can be appreciated. 9
2 Different types of answers in SQuAD 1.1 (Rajpurkar et al., 2016) dataset. . 17
3 Results of the BIDAF and BIDAF++ system in SQuAD 1.1 and SQuAD 2.0. 17
4 Distribution of domains from CoQA 22
5 The hyper-paramaters of the BERT pre-training and fine-tunning. Between brackets, the parameters proposed on the original paper. 35
6 Results obtained taking the context into account in the development set of the QuAC dataset as the test set is not publicly available. 36
7 Results obtained without taking the context into account in the development set of the QuAC dataset as the test set is not publicly available. 36
8 Statistics of the developed dataset compared with the QuAC and CoQA dataset. 46

Abbreviations

AMT Amazon Mechanical Turk

CNN Convolutional Neural Network

CQA Community Question Answering

DL Deep Learning

FAQ Frequently Asked Questions

IR Information Retrieval

MT Machine Translation

NER Named Entity Recognition

NLI Natural Language Inference

NLP Natural Language Processing

QA Question Answering

RNN Recurrent Neural Network

1 Introduction

Natural language is the most common interaction mode between humans and it is becoming increasingly important as a way to interact with machines too. As computer systems become more integrated into everyday tasks, the desire to make human-computer interaction as natural and efficient as possible is continuously growing. The development of systems that implement human conversation opens novel opportunities for human-machine interaction that go beyond the traditional devices like keyboards and that are applicable in many new situations. For example, hands-free and eye-free interaction can be applicable in car driving or home control applications. These novel communication systems suit perfectly with the new technological trends of the Internet of the Things and smart devices. Apart from that, recent advances in speech recognition have brought voice agents like *Amazon Alexa, Google Now, Siri* and *Cortana* and their use is becoming increasingly common. Amazon, for instance, sold more than 100 milion Alexa-powered devices in 2018 [1], opening high expectations for natural dialogues with machines. The incursion of these voice agents in our daily lives have brought a huge interest on companies to make information accessible by dialogue means. Traditionally, people have accessed Community Question Answering (CQA) forums or Frequently Asked Questions FAQ pages when seeking for answers for a specific question. However, this kind of information access is usually tedious since long text passages are retrieved as answers, so dialogue has been nominated as the solution for this issue.

Nowadays, conversational agents can be used to to check for landmarks, send web search queries and set an alarm using a few spoken words. However, the ability of these systems to keep a real and natural conversation is still very limited. One of the main reasons for this limitation is that in most commercial applications user utterances are processed with hand-designed rules, that can be reliable in certain domains, but inherently hard to scale to new domains as encoding all features a user might refer to in a conversation is an extremely challenging task. More recent approaches for dialogue involve training machine learning models on annotated training data. These techniques have shown to be much more effective than traditional rule based approaches in Natural Language Processing (NLP) tasks, specially the Deep Learning (DL) models that have dramatically overturned many NLP tasks: Natural Language Inference (NLI), Named Entity Recognition (NER) (Devlin et al., 2018), Machine Translation (MT) (Vaswani et al., 2017), etc. This has motivated a high interest in DL-based dialogue systems. However, as for dialogue systems, the neural revolution is still in its infancy, mainly due to the lack of labeled data, as machine learning techniques requires massive amounts of it.

[1] `https://www.digitaltrends.com/home/amazon-sold-more-than-100-million-alexa-devices/`

1.1 Motivation and goals

The main motivation behind this project is the above mentioned limitation of current commercial dialogue systems. This limitation, together with the huge demand of dialogue agents that society is experimenting, constraints the scope of current conversational agents, often below user expectations. Overcoming these limitations is an ambitious goal, which would require general artificial intelligence. So, taken into account the mentioned social and industrial interest in accessing FAQ and CQA websites information in a more intuitive way, we will focus on CQA dialogues, where the answers need to be retrieved from FAQ or CQA sites.

Given the scope of a master thesis, we set the following goals:

- The analysis of the main techniques for question answering and dialogue agents. As the main motivation is to go beyond the limits of the actual dialogue systems, we will put special attention on the state-of-the-art neural systems since these neural techniques have brought great improvements in a lot of different NLP tasks as mentioned before.
- The analysis of the most popular datasets in question answering and dialogue, analyzing the datasets themselves and the different methods for collecting them.
- The development of a dialogue system applying state-of-the-art techniques. This system should be tested on an existing dialogue dataset.
- The development of a domain-specific dialogue dataset in order to fight the problems neural networks face in domains with low amounts of labeled data. We selected the cooking domain, because of the LIHLITH project (see below).

This master thesis is framed inside of the Learning to Interact with Humans by Lifelong Interaction with Humans (LIHLITH) CHIST-ERA project. The main idea behind LIHLITH is to introduce a new lifelong learning framework for the interaction of humans and machines on specific domains with the aim of improving the quality of existing dialogue systems and lowering the cost of deployment in new domains. The first step in order to develop this framework is to generate a cooking domain dataset as it is a general knowledge domain that can ease the collection of dialogues.

2 Background

In this first chapter all the background knowledge used in the development of this project will be presented. This chapter is divided in three main sections: neural methods for NLP, where general deep learning techniques used in NLP are presented, neural question answering, where the main techniques for neural Question Answering (QA) are explained and neural dialogue systems, where the main neural dialogue systems and datasets are analyzed.

2.1 Neural Methods for NLP

In the last years the neural revolution has arrived to almost every research line inside the computer science community. Apart from NLP, great advances have been shown in computer vision, bio medicine, robotics, etc. There are four main reasons behind the success of these techniques.

The **first** reason is that these neural systems are directly based on the data. The traditional machine learning systems were based on features that had to be generated by feature engineering, requesting a great design effort for achieving meaningful features. Once with this meaningful features, traditional machine learning models were used in order to learn regression or classification systems. In contrast, the neural networks are capable of learning this abstract representation by themselves moving the feature engineering task from humans to machines. This property enables neural networks to learn all the latent features that can occur between the original inputs (x) and the final outputs (y) in and end-to-end way.

The **second** reason is the huge amount of available data we have nowadays thanks to the current information storage technology. Neural networks require huge amount of data in order to train and optimize their weights so they take great advantage of this data availability.

The **third** reason is the technological advance we have experienced on hardware. Neural networks are not as recent as we think, the first paper that uses the backpropagation of errors for training an artificial neural network was written in 1990 (Werbos, 1990). But until the advances on the processing power of the Graphical Processing Units (GPUs) the training of complex neural networks in reasonable time economically was very costly and time consuming.

The **fourth** and last reason is the vast amount of resources that are available in order to develop fast experiments and designs based on neural networks. This amount of resources mainly comes from the research community that makes available all the advances achieved using these techniques. Among these resources are worth mentioning all the open-source

libraries that ease the programming of neural networks: *TensorFlow*, *Torch*, *Keras*, etc.

Neural Networks come in various flavors, types and architectures. Describing all these types is beyond the scope of this project. Nevertheless, in the following paragraphs, we will briefly show the most widely used Neural Networks types:

Feed forward networks (see Figure 1) are the simplest neural networks that we can find. These networks learn to approximate an arbitrary function for mapping an input to an output, that could be a classification or a regression depending on the task. The learning of this arbitrary function is done by giving examples of the real data to the network that will adjust the weights of its neurons for optimizing an objective function. In this kind of networks all the connections between neurons are done in a forward direction.

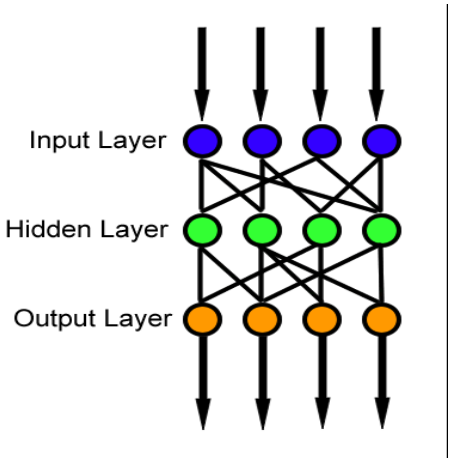

Figure 1: An example of a feed forward neural network. In this network the information from the input layer is moved and transformed through the hidden layer to the output layer. Source: Wikipedia, `https://en.wikipedia.org/wiki/Feedforward_neural_network`

Recurrent neural networks (RNN) (see Figure 2) are designed for processing sequences. They are based on the repetitive use of a basic neural network, where the output of each timestep is the input of the following one. They have the ability to keep information through time thanks to a memory. Depending on the memory management performed by the network we can define different recurrent network types: simple ones (Vanilla recurrent neural networks), LSTM (Long Short-Term Memory) networks (Hochreiter and Schmidhuber, 1997) and GRU (Gated Recurrent Unit) networks (Chung et al., 2014). While simple recurrent neural networks use basic mechanisms for updating the memory in each timestep, LSTMs and GRUs use advanced gated mechanisms for controlling the memory updates.

Convolutional neural networks (CNN) take their name from the mathematical convolution operation. In a convolutional operator a filter is applied over a signal. In

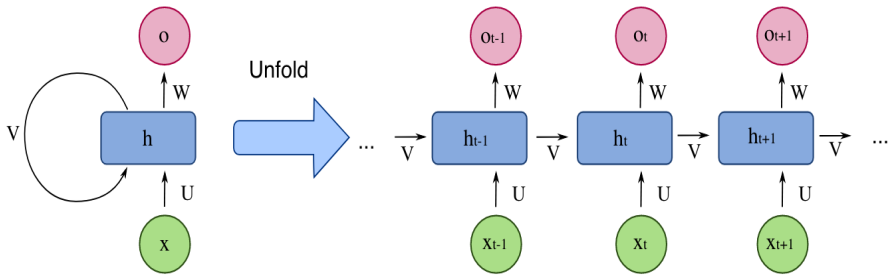

Figure 2: An example of a RNN. From bottom to top: input state (x), hidden state (h) and output state (o). U, W and V are the weights of the network. The unfolded version of the RNN can be seen in the right side. Source: Wikipedia, `https://en.wikipedia.org/wiki/Recurrent_neural_network`

the case of the convolutional neural networks, a neural network (the filter in this case) is applied to an input, achieving a feature map. Just like all the previously mentioned neural networks, the convolutional ones are able of adjusting the filter parameters by themselves, and this way they can extract interesting features from the input data. Convolutional operators are normally used before a pooling or subsampling layer, which enables the network to reduce the dimensionality of the input and extract the most meaningful features layer after layer (see Figure 3).

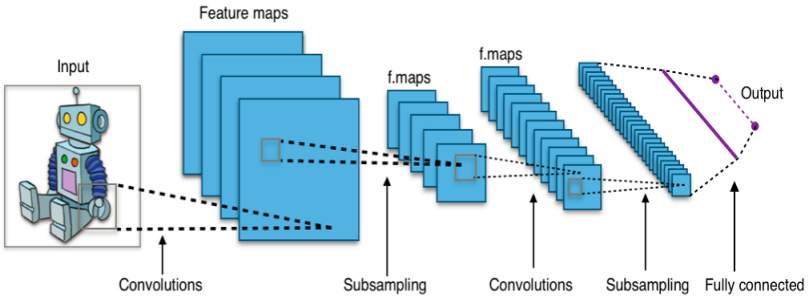

Figure 3: An example of a CNN where given an input image two convolution and subsampling layers are used. Source: Wikipedia, `https://en.wikipedia.org/wiki/Convolutional_neural_network`

So, due to all of the reasons mentioned before, NLP research community is slowly reducing the use of the machine learning methods that take advantage of features mined from syntax and text. Instead of these traditional models, complex systems built up from stacked neural networks, similar to those explained above, are the most common ones nowadays, together with **word embeddings** that are used as abstract representations of words.

Word embeddings are representations of words in the form of continuous vectors lying

in a multidimensional space, which are built by exploiting co-occurrence patterns extracted from corpora. As a consequence, vectors representing words that have similar meanings are close in the embedding space. In order to learn these vectors, large corpora are used and techniques as **C-BoW, Skip-Gram** (Mikolov et al., 2013) or **Glove** (Pennington et al., 2014) are applied. All these techniques take as a starting point the idea of similar words appearing in similar contexts, so they will mine the word embeddings from word co-ocurrences in large corpora. However, they use very different procedures for achieving these vectors:

In the case of C-BoW and Skip-gram, neural networks are used for learning language models (see Figure 4). C-BoW (Continuous Bag-of-Words) updates the distributional representation of a certain word taking into account the context words in a window. Differently, the Skip-gram model updates the distributional representation of the contextual words of the window taking into account the current word being processed.

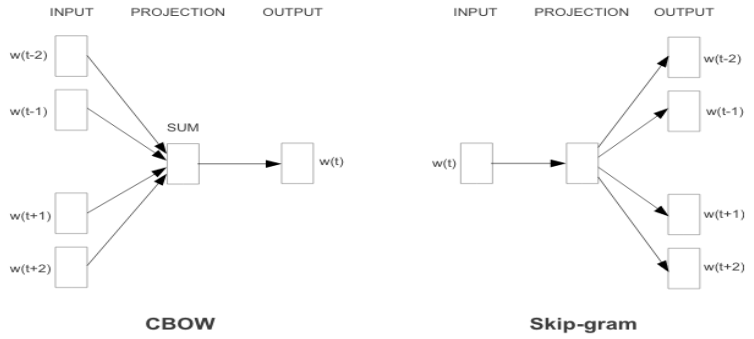

Figure 4: Visual representation of C-BoW and Skip-gram. Given context words $\{w(t-2), w(t-1), w(t+1), w(t+2)\}$ C-BoW sums their representations and tries to predict the target word $w(t)$. On the contrary, Skip-gram tries to predict context words $\{w(t-2), w(t-1), w(t+1), w(t+2)$ given $w(t)\}$. Source: (Mikolov et al., 2013)

On the other hand, we have the Glove model that reads the whole corpus and after building the co-occurrence matrix uses some optimization functions to the abstract representation of words.

Even if these word vectors are having great success in NLP they have some well-known issues. One of the most discussed problems is how to combine word embeddings to represent more complex segments of text, such as sentences. At this point, the scientific community is not sure if these fixed size scalar vectors are able to capture all the information that a variable length sentence contains. Usually, sentence representations are computed using recurrent neural networks such as LSTM and biLSTM (Cho et al., 2014b). However, in the last year several authors propose using **attention mechanisms** (Bahdanau et al., 2014)

to address this issue.

Attention mechanisms are functions that map two input vectors into a scalar output. This scalar output measures and encodes the interaction between the two inputs and it is computed as a weighted sum of the elements in both vectors. In the case of NLPg attention mechanisms are used in order to model the interaction between words and have been specially used in machine translation tasks (Bahdanau et al., 2014).

Lately, attention mechanisms have been proposed as an alternative to the traditional recurrent neural networks explained before. The authors in (Vaswani et al., 2017) claim that the **Transformer** architecture, that is a feed forward network based solely on attention mechanisms, is able to outperform the traditional recurrent neural networks in less training time as not recurrence is found on them. The Transformer uses self-attention mechanisms together with a positional encoding in order to model the sentence order.

Another weak point of word embeddings is the difficulty for representing the polisemy and the polarity of words. For example, the words "pretty" and "ugly" will appear in similar contexts, since when learning the embeddings only small windows of the context are used, and this will end up with antonym words having similar abstract representations. With the intention of finding a solution to this problem, the idea of **deep contextualized word representations** was proposed. So, in this case, instead of having a single abstract representation for each word, we will have different representations for each word depending on the context it appears. The most important systems that implement these contextualized word representations are **ELMo** (Peters et al., 2018) and **BERT** (Devlin et al., 2018). ELMo uses two recurrent neural networks, one in left-to-right direction, and the other in right-to-left direction, and at the end it concatenates the two abstract representations of the left and right contexts of the processed word. On the other hand, BERT learns contextualized embeddings in a joint way for the left and right contexts taking advantage of the Transformer architecture.

2.2 Neural Question Answering

We start asking questions and seeking answers for exploring and making sense of our surroundings since childhood (Vale, 2013). So, it is not surprising that when computers appeared we started asking questions to them, making QA one of the most traditional computer science tasks. This long tradition has ended up in great advances in QA. It is worth mentioning the case of IBM's Watson system, that in 2011 won the TV game-show *Jeopardy!* surpassing humans capacity at answering questions like the following one:

The author of the popular song *Gernikako Arbola* was born in this village located in Gipuzkoa. [2]

[2]This is just a similar question to the questions of the show. The answer for this is Urretxu (Gipuzkoa).

This system is specially well-known due to the media impact of a computer beating humans in intelligent tasks, but current systems tend to focus on **factoid questions** instead. These questions are answerable with simple facts and expressed in short texts. Figure 5 shows an example of a factoid question from the SQUAD 1.1 dataset (Rajpurkar et al., 2016).

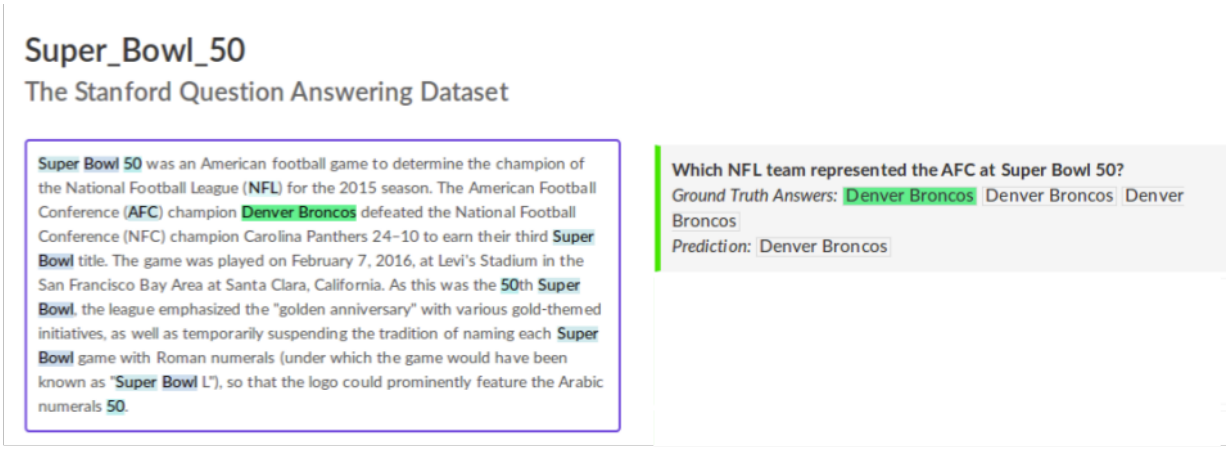

Figure 5: An example from the SQUAD 1.1 dataset where a factoid question is shown with the ground truth answer spotted in the text. (Rajpurkar et al., 2016)

2.2.1 Methods for Question Answering

There are a wide variety of ways for classifying QA systems (Mishra and Jain, 2016), but we will follow the one stated in (Jurafsky and Martin, 2014). According to the authors, traditionally there have been two major paradigms for QA: **knowledge based** and **information retrieval based** models. There are also what they call hybrid methods that find answer candidates using both techniques and then rank them for choosing the most suitable one.

Knowledge-based question answering

In knowledge-based QA the task is to map a natural language question to a query over a structured knowledge-base. Usual databases are full **relational databases**, queried using SQL language, or simpler ones like **RDF triplets**, that are queried using SPARQL. In order to tackle this task, **rule-based** and **supervised methods** have been used. Rule-based methods are specially useful for extracting very frequent relations like:

"Where was Jose Mari Iparragirre born?" → birth-place (Jose Mari Iparragirre, ?x)

The problem with this type of **rule-based methods** is that they are **not scalable** for non frequent relations, since it is impossible for humans to hand code and predict all the

Question	SQL query
What is the number of cars with more than 4 cylinders	SELECT COUNT(*) FROM cars_data WHERE cylinders >4
For each stadium, how many concerts are there?	SELECT T2.name, COUNT(*) FROM concert AS T1 JOIN stadium AS T2 ON T1.stadium_id = T2.stadium_id GROUP BY T1.stadium_id
Which countries in Europe have at least 3 car manufacturers?	SELECT T1.country_name FROM countries AS T1 JOIN continents AS T2 ON T1.continent = T2.cont_id JOIN car_makers AS T3 ON T1.county_id = T3.county WHERE T2.continent = 'Europe' GROUP BY T1.country_name HAVING COUNT(*) >= 3

Table 1: Some examples from the SPIDER dataset (Yu et al., 2018b). In the left column the text question is given and in the right column the SQL language mapping can be appreciated.

possible relations.

On the other hand, the supervised methods take advantage of a set of questions paired with their logical or query language form like in Table 1. Then, the task is to build a system that given a question is able to map it into its logical or query language form. In order to build reliable supervised systems, we need large-scale datasets of input-output pairs, like the **SPIDER** text-to-SQL dataset (Yu et al., 2018b), that contains **10,181 human-tagged questions**. Another interesting characteristic of this dataset is that it covers **cross-domain queries** from 138 different domains and 200 databases. The best performing system called **SyntaxSQLNet** (Yu et al., 2018a) achieves 19.7 of exact match in the test set. This means that only 19.7% of the predicted queries are equivalent to the gold query as a whole .

Information-retrieval based question answering

The previously mentioned knowledge-based systems use structured data for answering questions. Using structured data for this task allows to build quite reliable systems. But, as the amount of information stored in structured ways is limited, the idea of **information-retrieval based QA** comes up. **Information-retrieval** or **IR-based** QA has two main steps. In the first one, given a user question IR techniques are used to collect relevant documents and passages from the web. There are four main IR model types: set-theoretic

models, algebraic models, probabilistic models and feature based models (Baeza-Yates et al., 2011). Although these models are broadly used in current commercial systems, we are not going to go deeper on them since they are not used in this thesis project.

The second step of IR based QA consists of using **machine comprehension** algorithms for understanding the content of the relevant documents and trying to answer the user questions afterwards. In this thesis project we will specially focus on this second step that is also known as **Answer Extraction**.

Answer Extraction

In answer extraction the relevant documents for the user question will be given to us as input. In other words, we will be provided with a document or passage p of M tokens $\{p_1, p_2, ..., p_m\}$ and a user question q of N tokens $\{q_1, q_2, ..., q_n\}$ and we will have to find an answer a that could be generated (abstractive) or just extracted (extractive) from the text by computing the start $p_{start}(i)$ and end $p_{end}(i)$ probabilities of each token i in p. $p_{start}(i)$ will represent the probability of token i for being the starting token of the answer. In a similar way, $p_{end}(i)$ represents the probability of token i for being the ending token of the answer. Nowadays the extractive approach is the most popular one as it makes the evaluation of the systems much easier.

The usual strategy used to address this task is using **supervised machine learning** techniques, in which we have to infer a function that maps the input to an output based on some **training data**. This training data will be composed by some input-output pairs, see Figure 6 for an example.

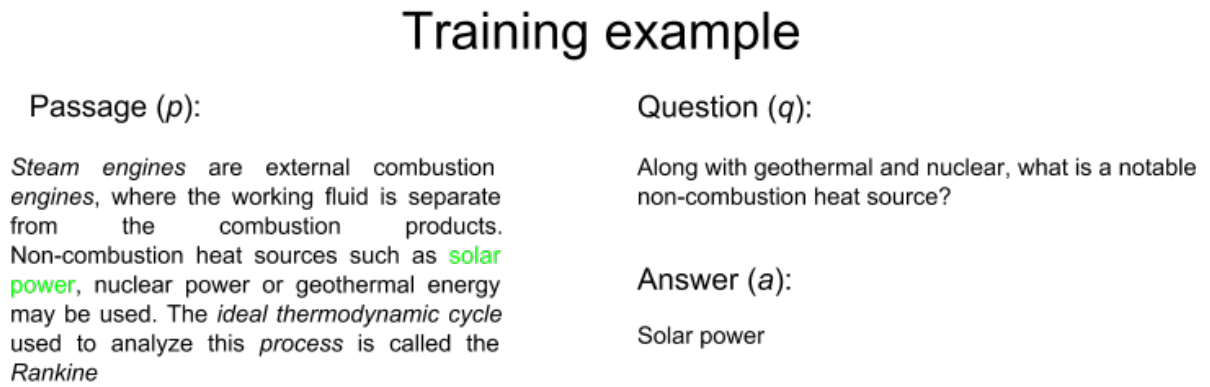

Figure 6: A training example taken from SQUAD 2.0 dataset (Rajpurkar et al., 2018). Here, the passage p, the question q and the answer a, which is spotted in the text, can be appreciated.

Like in almost any NLPg task, the **feature-based methods**, in which classifiers were trained with hand crafted features have been surpassed by **neural network** approaches.

2.2.2 Neural Question Answering Systems

After giving a general overview of the types of QA systems, we will step into the state-of-the-art systems used on the development of this thesis. As mentioned in section 2.2.1, we will just focus on the second step of information-retrieval based QA, that consists of answering a query about a given document or context paragraph, also known as **machine comprehension**.

Bi-Directional Attention Flow

Bi-Directional Attention Flow for Machine Comprehension or BIDAF [3] (Seo et al., 2016) is an extractive answer extraction neural system that uses a multi-stage hierarchical process for representing the context at different levels of granularity. However, the most special feature of this system is the use of the bi-directional attention mechanism that substitutes the traditional method of summarizing the passage to fixed size vectors with uni-directional attention.

The overview of the model can be seen in Figure 7. The hierarchical multi-stage process consists of the following six layers:

1. **Character Embedding Layer** that uses character-level Convolutional Neural Networks as in (Kim, 2014) for mapping words of length j to a high-dimensional vector space. First of all, characters are embedded into vectors of length k that are then given to the CNN. The outputs of this network are then max-pooled to obtain a fixed size vector. An example of this type of CNNs can be seen in Figure 8.

2. **Word Embedding Layer**. The main idea behind this layer is the same of the previous one. Pre-trained GloVe embeddings (Pennington et al., 2014) are used for mapping each word to a high-dimensional vector-space.

   Once both vector representations are achieved, the one obtained with the character-level Convolutional Neural Network and the other from the pre-trained GloVe embeddings, the concatenation of both vectors is passed to a two-layer Highway-Network (Srivastava et al., 2015). This Highway-Network is used in order to decide if information is kept or transformed for upper layers. The output of this network consist of two matrices: one for the passage $P \in \mathbb{R}^{d \times M}$ and one for the question $Q \in \mathbb{R}^{d \times N}$, being d the size of the output vectors, M the size of the passage and N the size of the question.

[3] Code for BIDAF is available at: `https://github.com/allenai/bi-att-flow`

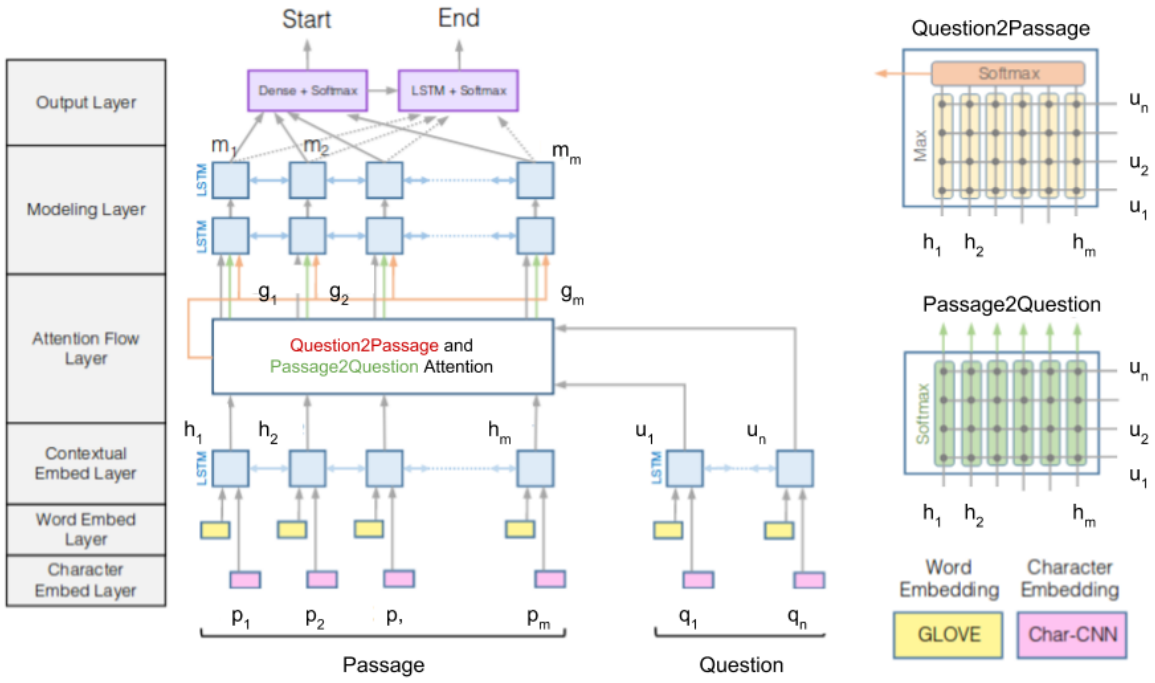

Figure 7: Architecture of the BIDAF system. In the left side of the picture the six main layers of the system can be appreciated. Source: (Seo et al., 2016)

3. **Contextual Embedding Layer**. In order to model interactions between words, a bi-directional Long Short-Term Memory Network (Hochreiter and Schmidhuber, 1997) is used. This bi-directional LSTM consists of two uni-directional LSTMs, one in left-to-right direction and the other in right-to-left direction. At the end of the process their outputs are concatenated. So after using the bi-LSTM two new matrices are achieved: $H \in \mathbb{R}^{2d \times M}$ for the passage and $U \in \mathbb{R}^{2d \times N}$ for the question. Notice that each column of the matrices is $2d$ dimensional due to the concatenation of the forward and backward directional LSTMs.

4. **Attention Flow Layer**. This layer is used for fusing the information computed from the passage and question. Instead of using the attention layer for combining the passage and query information into a single size feature vectors, this method proposes to let the attention vectors and the embeddings from previous layers flow through the following layers. This way, the information loss due to early summarization is avoided.

   So given the H and U matrices from the previous layer, the attention-flow layer will compute a matrix G with the question-aware vector representations of the context words.

   In order to compute this question aware vector representations, attention is computed

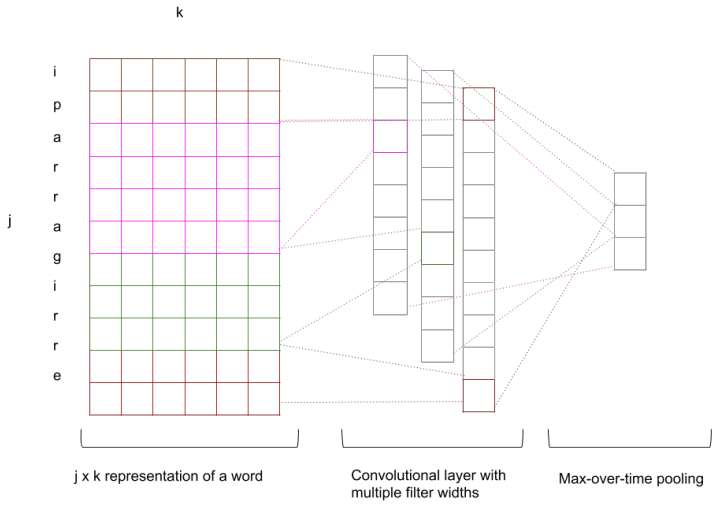

Figure 8: An example of a character based CNN for mapping the word "Iparragirre" to high-dimensional vector space.

in both directions: from context to query and from query to context. This bi-directional attention is derived from a matrix $S \in \mathbb{R}^{M \times N}$, where S_{mn} expresses the similarity between passage word m and question word n. This matrix is computed by the following formula:

$$S_{mn} = \alpha(H_{:m}, U_{:n}) \in \mathbb{R} \tag{1}$$

α is a trainable scalar function defined in the following way : $\alpha(h, u) = w_s^\intercal[h; u; h \circ u]$, where $w_s \in \mathbb{R}^{6d}$ consists of a trainable weight vector, $\circ$ is element wise multiplication and $[;]$ is vector concatenation. $: m$ and $: n$ refer to the mth and nth rows of H and U matrices respectively.

Passage-to-question Attention (P2Q). This attention mechanism represents which are the most relevant question words for each passage word. So, $a_m \in \mathbb{R}^n$ represents the attention weights on the question words for the m-th passage word, where $\sum a_{mn} = 1$. This attention weight is computed by a softmax function: $a_m = softmax(S_{m:}) \in \mathbb{R}^N$. Once the weight is computed for each passage word, the attended vectors are computed in the following way: $U'_{:m} = \sum_j a_{mj} U_{:j}$. We end up with U', a $2d$-by-M matrix that contain attended vectors for all the passage words.

Question-to-passage Attention (Q2P). Question-to-passage attention represents just the opposite aspect from the previous attention mechanism. In this case, the attention mechanism is used to represent which passage words are more closely related with the question words, being these words critical for answering the question. Attention weights on the passage words are obtained with $b = softmax(max_{col}(S)) \in \mathbb{R}^M$, where the maximum function is performed across the column. Then the attended vector is computed by $h' = \sum_m b_m H_{:m} \in \mathbb{R}^{2d}$, that will indicate the weighted sum

of the most important passage words with respect to the query. Then this vector is expanded M times across the columns to obtain a matrix $H' \in \mathbb{R}^{2d \times M}$.

At the end of this layer, the contextual embeddings and the attention vectors are combined in the following way:

$$G_{:m} = \beta(H_{:m}, U'_{:m}, H'_{:m}) \in \mathbb{R} \tag{2}$$

Where β is defined by a simple concatenation: $\beta(h, u', h') = [h; u'; h \circ u', h \circ h'] \in \mathbb{R}^{8d \times M}$.

5. **Modeling Layer.** The output of the Attention Flow Layer is now passed to a two layered bi-directional LSTM. After using this bi-LSTM we achieve a matrix $M \in \mathbb{R}^{2d \times M}$ that captures interaction among the question conditioned passage words.

6. **Output Layer.** As mentioned in section 2.2.1, the aim of the extractive QA systems is to compute the start p_{start} and end p_{end} probabilities of each token in the passage. The start probability is achieved with the following formula:

$$p_{start} = softmax(w^{\intercal}_{(p^{start})}[G; M]) \tag{3}$$

where $w^{\intercal}_{(p_{start})} \in \mathbb{R}^{10d}$ is a trainable weight vector. For computing the end probability, M is passed through another bi-LSTM layer to obtain $M^2 \in \mathbb{R}^{2d \times M}$. This new matrix in then used in the following way:

$$p_{end} = softmax(w^{\intercal}_{(p^{end})}[G; M^2]) \tag{4}$$

The training loss of this model is defined as the sum of the negative log probabilities of the true start and end indices and the predicted ones, averaged over all examples:

$$L(\theta) = -\frac{1}{N} \sum_{i}^{N} log(p^{start}_{y_i^{start}}) + log(p^{end}_{y_i^{end}}) \tag{5}$$

where θ are all the trainable parameteres in the model, N is the dataset number of examples and y_i^{start} and y_i^{end} are the true labels of the i-th example.

Extended Bi-Directional Attention Flow

Extended Bi-Directional Attention Flow or BIDAF++ is a modification of the BIDAF system proposed in (Clark and Gardner, 2017) that also substitutes pre-trained GloVe word embeddings with ELMo contextualized word representations (Peters et al., 2018). The architecture of the modified system can be seen in figure 9.

The main difference proposed in this extended model is the addition of a **Self-Attention Layer** after the Attention Flow Layer of the original system. Together with that, all

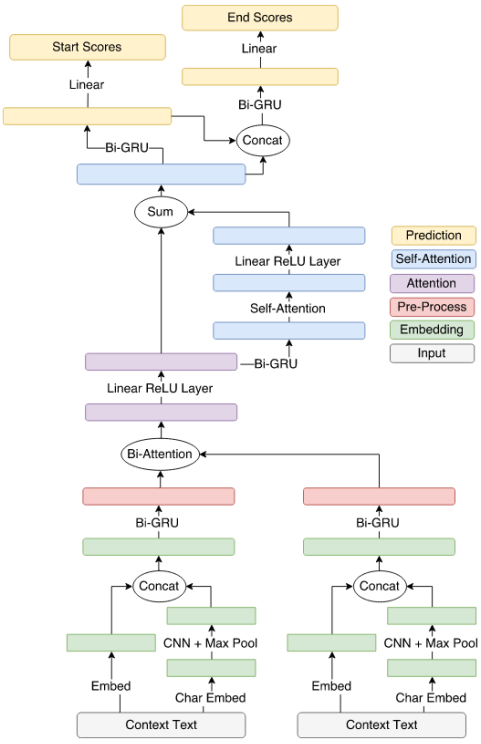

Figure 9: Architecture of the extended BIDAF++ system. Source: (Clark and Gardner, 2017)

the LSTMs are substituted by gated recurrent units (GRUs) (Cho et al., 2014a) and the highway-networks are removed from the Embedding Layer.

Self-Attention Layer that is composed by the blue coloured sub-layers in Figure 9. The input of this layer is the output of the Attention Flow Layer of the BIDAF system after passing through a linear layer with ReLU activations. The first step inside this Self-Attention layer is to apply a bi-directional GRU. Then, the same attention mechanism of the original system is applied, but between the passage and itself and without the question-to-passage attention. The new attended vectors are passed again though a linear layer with ReLU activations and then concatenated with the input of the layer as this is a residual layer.

The remaining layers of the model remain intact except for the Prediction Layer, named as Modelling Layer in original BIDAF, where one of the two bi-GRUs is removed.

2.2.3 Question Answering Datasets

There is a large set of QA datasets available, but in this section we will focus on the ones that have served as inspiration for the dialogue dataset developed in this project.

SQuAD 1.1

The Stanford Question Answering Dataset (SQuAD) (Rajpurkar et al., 2016) is an extractive QA dataset that consists of more than 100,000 questions proposed by crowdworkers from Wikipedia articles. The process followed for collecting the dataset is divided in three stages: passage curation, question-answers collection and obtaining additional answers.

1. **Passage curation.** In this stage they take 536 randomly sampled articles from the top 10000 ranked ones. After discarding images, figures, tables and articles shorter than 500 characters they end up with 23,215 paragraphs from the original 536 articles.

2. **Question-answer collection.** This collection is completed using Amazon Mechanical Turk (AMT) as its backend. How AMT works will be further explained in section 4.2. The crowdworkers were required 97% HIT acceptance rate, a minimum of 1000 HITs, and to be USA or Canada residents. Workers were paid $9 per hour and asked to spend 4 minutes on every paragraph.

   The task itself consisted of asking and answering up to 5 questions on the content of each paragraph. The questions are free text and have to be answered with a span of text highlighted in the paragraph. In order to encourage crowdworkers to use their own words when asking, a reminder was shown at the beginning of every paragraph and copy/paste function was disabled.

3. **Additional answers collection.** In order to evaluate human performance and make evaluation more robust, the authors obtained two additional answers for each question. These new answers were achieved by showing to crowdworkers the questions that another worker had previously done. Then the task was to answer these questions again with the shortest possible span of text. The amount of paid money in this case remains the same, $9 per hour.

The different types of answers in this dataset can be seen in Table 2.

SQuAD 2.0

SQuAD 2.0 [4] (Rajpurkar et al., 2018) is an enriched version of the previous SQUAD 1.1 dataset in which unanswerable questions are added. The motivation for adding this kind of questions is the great difficulty that reading comprehension systems face when the answer to the question does not appear in the text. This new dataset is formed by the previous SQuAD 1.1 dataset combined with over 50,000 new unanswerable questions.

In order to achieve relevant and plausible questions on the topic, crowdworkers are asked to pose up to five questions that are impossible to answer for each paragraph in the original SQuAD 1.1 dataset. As inspiration and motivation for getting similar questions

[4] SQuAD 2.0 dataset is available at: `https://rajpurkar.github.io/SQuAD-explorer/`

Answer Type	Percentage	Example
Date	8.9 %	18 November 1995
Other Numeric	10.9 %	23
Person	12.9 %	Jose Mari Iparragirre
Location	4.4 %	Urretxu
Other Entity	15.3 %	EHU
Common Noun Phrase	31.8 %	Master Thesis
Adjective Phrase	3.9 %	second-largest
Verb Phrase	5.5 %	returned to Earth
Clause	3.7 %	to avoid trivialization
Other	2.7 %	quietly

Table 2: Different types of answers in SQuAD 1.1 (Rajpurkar et al., 2016) dataset.

to original ones, examples from the previous dataset are shown to workers. Workers are asked to spend 7 minutes in each paragraph, and paid $ 10.50 per hour.

The results obtained by the **BIDAF** and **BIDAF++** systems explained in section 2.2.2 can be seene in Table 3.

	SQuAD 2.0		SQuAD 1.1	
System	Exact match	F1	Exact match	F1
BIDAF	59.17	62.09	67.97	77.32
BIDAF++ (Glove)	59.33	62.30	72.13	81.04
BIDAF++ (ELMo)	**63.37**	**66.25**	**78.66**	**85.83**

Table 3: Results of the BIDAF and BIDAF++ system in SQuAD 1.1 and SQuAD 2.0.

2.3 Neural Dialogue Systems

The **Turing test** (Turing, 1950) tries to test the ability of a machine to show intelligent behaviour equivalent to that of a human. This test consists of a human evaluator judging natural language dialogues between a human and a machine designed to generate human like responses. The human is aware of the fact that one of the participants is a machine and the other not. At the end of the conversation if the human is not able to claim which of the participants is the machine, this machine is supposed to be intelligent. In Figure 10 the standard interpretation of this test can be appreciated.

This test is one of the most well known tests for testing intelligence on machines and dialogue takes an important part on it, so since the proposal of the Turing test in 1950 the development of **dialogue systems** or **conversational agents** has been a main research topic on computer science.

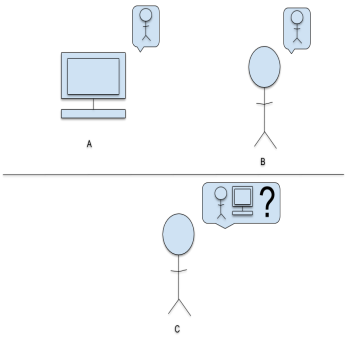

Figure 10: Standard interpretation of the Turing test (Turing, 1950)

2.3.1 Methods for Dialogue

Dialogue systems are usually divided in two different groups: **task-oriented dialogue systems** and **chatbots**.

Task-oriented dialogue systems

Task oriented dialogue systems are built for a particular task and usually designed to have short conversations. All the trendy digital assistants that are now implemented in mobile phones or home controllers like *Siri, Cortana, Alexa, Google Now...* are inside of this group of task-oriented conversational agents and they have increased the research interest on this type of systems due to the market success they are experiencing. These conversational agents are able to give travel directions, control smart homes appliances, recommend restaurants, give information about your favourite sports... Companies usually implement task-oriented agents in their websites too in order to access on the customer's frequently ask questions (FAQs).

Nowadays, the commercial goal-oriented systems process user utterances with hand-crafted **semantic frames** that contain a collection of **slots** and define the **values** each slot can take. Slot-filling pre-defines the structure of a dialogue as a set of slots to be filled during the dialogue turns. Even if this technique has proven to be reliable, it has the problem of not scaling well to new domains as manually encoding all the slots a user might refer to in a conversation is a extremely challenging task.

Another problem with this frame-based system is that usually they do not take into account the intention of the person with whom they are having interactions. As it can be appreciated in Figure 11, where baseball player names have been invented, knowing if the other person is making a statement or asking questions is very important in order to carry out a good dialogue. The list of actions that represent the intentions behind a dialogue will be named after **dialogue acts**. An example of some dialogue acts can be seen in figure

12.

C: I want you to tell me the names of the fellows on the St. Louis team.
A: I'm telling you. Who's on first, What's on second, I Don't Know is on third.
C: You know the fellows' names?
A: Yes.
C: Well, then, who's playing first?
A: Yes.
C: I mean the fellow's name on first.
A: Who.
C: The guy on first base.
A: Who is on first.
C: Well what are you askin' me *for?*
A: I'm not asking you – I'm telling you. Who is on first.

Figure 11: A conversation about invented baseball players. Source : (Jurafsky and Martin, 2014)

Tag	Sys	User	Description
HELLO($a=x,b=y,...$)	✓	✓	Open a dialog and give info $a=x,b=y,...$
INFORM($a=x,b=y,...$)	✓	✓	Give info $a=x,b=y,...$
REQUEST($a,b=x,...$)	✓	✓	Request value for a given $b=x,...$
REQALTS($a=x,...$)	χ	✓	Request alternative with $a=x,...$
CONFIRM($a=x,b=y,...$)	✓	✓	Explicitly confirm $a=x,b=y,...$
CONFREQ($a=x,...,d$)	✓	χ	Implicitly confirm $a=x,...$ and request value of d
SELECT($a=x,a=y$)	✓	χ	Implicitly confirm $a=x,...$ and request value of a
AFFIRM($a=x,b=y,...$)	✓	✓	Affirm and give further info $a=x,b=y,...$
NEGATE($a=x$)	χ	✓	Negate and give corrected value $a=x$
DENY($a=x$)	χ	✓	Deny that $a=x$
BYE()	✓	✓	Close a dialog

Figure 12: Some examples of dialogue acts. Source : (Jurafsky and Martin, 2014)

Systems that understand the dialogue acts usually follow a modular design that is known as the **dialog-state** architecture (see Figure 13). The main difference the dialog-state architecture has is the usage of a **dialog state tracker** and a **dialog policy**. The first one will maintain the current state of the dialog and the second one will decide what should the system do next.

Chatbots

Chatbots are systems that mimic human-human interactions and carry on extended non task specific conversations. Opposed to task-oriented dialogue systems, chatbots are usually designed for entertainment. In a very similar way to QA defined in section 2.2.1, they

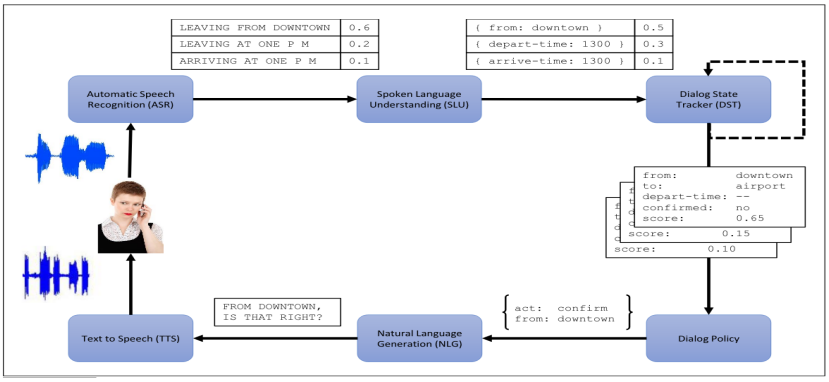

Figure 13: Architecture of a dialogue-state system. Source : (Jurafsky and Martin, 2014)

are also divided in two main groups: **rule-based** chatbots and **corpus-based chatbots**.

The already well known issue of the difficulty for scaling rule-based systems is the main motivation for the usage of corpus-based chatbots.

Corpus-based chatbots mine conversations from human-human or human-machine conversation from available corpora such as Twitter or chat platforms. There are two main architectures inside of this group: **information retrieval chatbots** and **machine learned sequence transduction chatbots**. The first type take advantage of information retrieval algorithms for given a question q selecting the most similar response from a big corpus. Differently, Sequence to sequence (seq2seq) chatbots generate dialogue transducing the user input question into the system output answer. At the beginning systems from seq2seq machine translation were applied for response generation, however, it quickly became clear that this task was far from machine translation. In machine translation source and target sequences tend to align well with each other, but in conversation, a user question may have no words in common with a coherent response. This difficulty motivated the idea of generating dialogue by spotting answers in text in a similar way to the QA datasets explained in subsection 2.2.3. Once having these datasets QA techniques can be applied for chitchatting.

2.3.2 Neural Dialogue Systems

There are a wide range of neural dialogue systems that implement each of the modules inside the dialog-state architecture separately. However, we are interested on developing a system that is able to predict the dialog act and to spot an answer in a passage in an end-to-end manner. Due to this, we are just focusing on one neural dialogue system known as **BIDAF++ for dialogue**.

The neural dialogue system we are analyzing in this section is a mixture of task-oriented dialogue systems and chatbots. This duality comes from the usage of machine comprehen-

sion techniques for response spotting in passage and dialogue act prediction. So we can define our task in the following way:

Given a document or passage p of M tokens $\{p_1, p_2, ..., p_m\}$, a history of user questions and answers $Q_1, A_1, Q_2, A_2, ..., Q_{k-1}, A_{k-1}$ and a dialogue act set d of L discrete statements $\{d_1, d_2, ..., d_l\}$, we will have to find the answer A_k to question Q_k that will consist of the starting index i and ending index j of the answer on the passage together with a dialogue act list v.

BIDAF++ for dialogue

BIDAF++ for dialogue [5] (Choi et al., 2018) is a modification of the BIDAF++ system for QA, defined in section 2.2.2, in order to make it able to solve the just mentioned task. This modification will consist of adding a classifier over the same hidden representation used to predict the end position of the predicted span.

Apart from that, as BIDAF++ is designed just for QA and in this case they are working with dialogue, the authors also try to model the dialog context by modifying the passage and question embedding processes. In order to model the dialog history they consider the context from the previous N question and answer pairs: $Q_{k-N}, A_{k-N}, ..., Q_{k-1}, A_{k-1}, Q_k$. Each of these question and answer pairs are defined as a **dialogue turn**.

They modify the **passage embeddings** by concatenating embeddings from the previous N answers to the existing passage word embeddings. **Question embeddings** are also modified by encoding the dialog turn number within the question embedding. This additional embeddgins are marked with marker embeddings for avoiding unexpected behaviour from the system. For example, if the embeddings for the history are not marked after the concatenation with the passage embeddings, system could spot the answer on the context history and not on the original passage.

2.3.3 Dialogue Datasets

In this section the dialogue datasets that have motivated the developed dataset in this project will be presented.

CoQA

CoQA [6]is a dataset for building Conversational Question Answering systems proposed by (Reddy et al., 2018). This dataset contains 127k questions with answers, obtained from

[5]BIDAF++ for dialogue is available at: `https://github.com/allenai/allennlp/blob/master/allennlp/models/reading_comprehension/dialog_qa.py`

[6]CoQA dataset is available at: `https://stanfordnlp.github.io/coqa/`

8k conversations about passages from seven different domains. In Table 4 the distribution of the domains can be appreciated. The Science and Reddit domains are reserved for out-of-domain evaluation of the systems. Passages that have only one entity are removed for avoiding the dialogue to focus only on that entity. Apart from that, long articles are truncated to the first 200 words.

Domain	#Passages	#Q/A pairs	Passage length	#Turns per passage
Childen's Stories	750	10.5k	211	14.0
Literature	1,815	25.5k	284	15.6
Mid/High School exams	1,911	28.6k	306	15.0
News	1,902	28.7k	268	15.1
Wikipedia	1,821	28.0k	245	15.4
Out of domain				
Science	100	1.5k	251	15.3
Reddit	100	1.7k	361	16.6
Total	8,399	127k	271	15.2

Table 4: Distribution of domains from CoQA

The collection of the dataset is done following an interactive task between two annotators, a **questioner** and an **answerer**. The usage of two annotators helps to achieve more natural dialogues, as self chats tend to be quite artificial. The interface used for pairing the workers is Amazon Mechanical Turk and the cost of each passage is of $3.6 for conversation collection. In order to select the workers a qualification test that assesses their understanding of the guidelines is passed. This test is only done by turkers from english speaking countries with more than 1000 HITs and at least 95% acceptance rate. The qualification test was successfully completed by 547 workers out of 960.

The names of the two roles in this interactive task are quite self explanatory. The questioner will have to do questions given a passage from one of the previously mentioned domains (see Figure 14 for an example of the questioner's interface). In order to increase the lexical diversity in the questions, the questioner will be motivated to use different words from the ones that appear in the passage by receiving an alert to paraphrase them. On the other hand, the answerer will have to give a free-text answer after highlighting the span of text in the passage used as inspiration for the answer (see Figure 15 for an example of the answerer interface). For limiting the number of possible answers, the authors want the answerer to use similar vocabulary to the one of the passage. This attitude is encouraged by automatically copying the highlighted text to the answer box. CoQA authors claim that 78% of the answers had at least one edit, but as proven in (Yatskar, 2018) a 97.8 F1 upper bound can be obtained by an extractive system, showing that the changes should be minimal in almost all the cases.

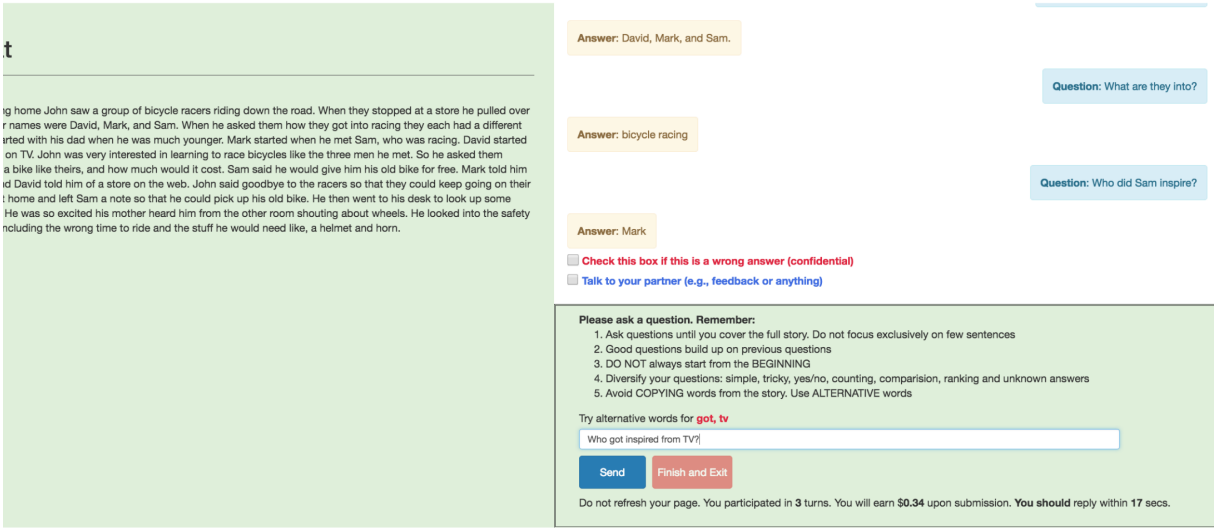

Figure 14: The questioner interface from CoQA. Source : (Reddy et al., 2018)

QuAC

QuAC [7](Choi et al., 2018) is a dataset that contains 14k information-seeking question answering dialogues. The collection of this dataset is done by an interactive task between two crowdworkers in which each of them carries out a different role. The used platform for this collection is Amazon Mechanical Turk restricting the task to workers in English-speaking countries with more than 1000 HITs and at least 95% acceptance rate. To ensure quality on the dialogues the authors use a qualification task that 278 from 645 workers passed. On average the cost of each dialogue was of $2.68.

This task pairs two workers, one with the **teacher** role and the other with the **student** role, in order to discuss a section from a Wikipedia article. A visual representation of the task can be appreciated in Figure 16. The student will have access to the section's title and the first paragraph of the main article as inspiration for formulating free-text questions. On the other hand, the teacher will have additional access to the full section text and will have to answer to the student's question selecting a contiguous span of text defined by two indices in the section. Apart from the span of text the teacher must also provide the student a list of dialog acts that are grouped in the following way:

- Continuation. It is used for leading the student to the most interesting topics: *follow up*, *maybe follow up* or *don't follow up*.
- Affirmation. It is required when the question is a Yes/No question: *yes*, *no* or *neither*.
- Answerability. It will define if the question has an answer or not: *answerable* or *no answer*.

[7]QuAC dataset is available at: `http://quac.ai/`

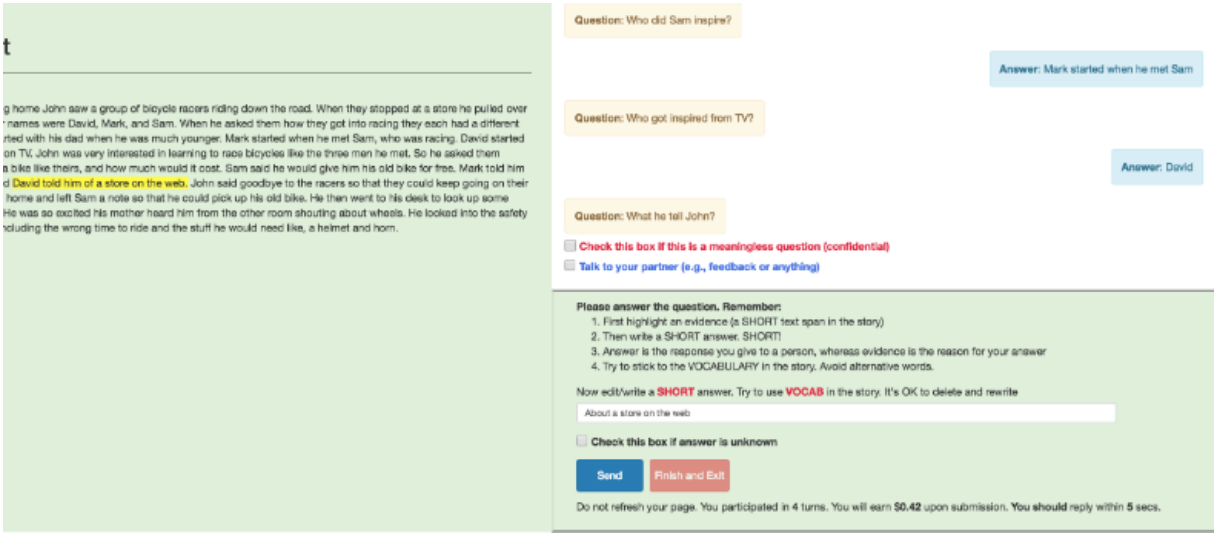

Figure 15: The answerer interface from CoQA. Source : (Reddy et al., 2018)

Finally, dialogues are ended either when they reach a maximum of 12 turns or when 3 unanswerable questions have been asked. An example of the collected dialogues can be seen in Figure 17.

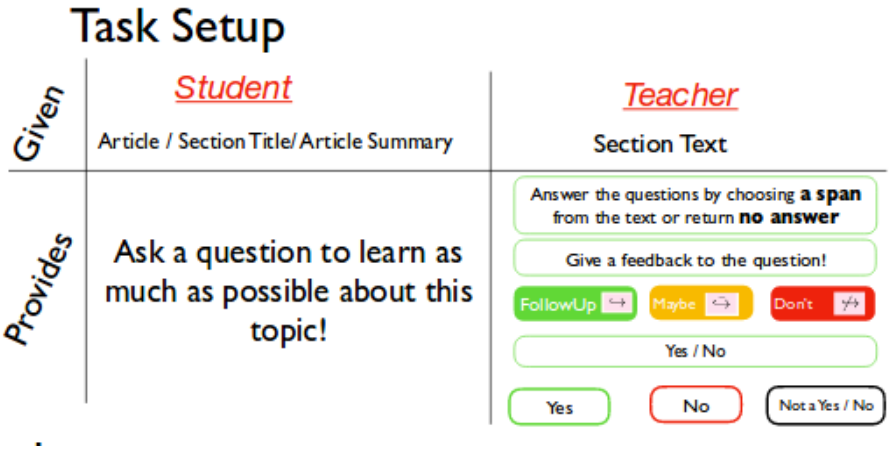

Figure 16: Visual representation of the interactive task proposed in QuAC. Source: poster of the QuAC dataset.

Section: Daffy Duck, Origin & History

STUDENT: **What is the origin of Daffy Duck?**
TEACHER: ↪ first appeared in Porky's Duck Hunt
STUDENT: **What was he like in that episode?**
TEACHER: ↪ assertive, unrestrained, combative
STUDENT: **Was he the star?**
TEACHER: ↪ No, barely more than an unnamed bit player in this short
STUDENT: **Who was the star?**
TEACHER: ↛ No answer
STUDENT: **Did he change a lot from that first episode in future episodes?**
TEACHER: ↪ Yes, the only aspects of the character that have remained consistent (...) are his voice characterization by Mel Blanc
STUDENT: **How has he changed?**
TEACHER: ↪ Daffy was less anthropomorphic
STUDENT: **In what other ways did he change?**
TEACHER: ↪ Daffy's slobbery, exaggerated lisp (...) is barely noticeable in the early cartoons.
STUDENT: **Why did they add the lisp?**
TEACHER: ↪ One often-repeated "official" story is that it was modeled after producer Leon Schlesinger's tendency to lisp.
STUDENT: **Is there an "unofficial" story?**
TEACHER: ↪ Yes, Mel Blanc (...) contradicts that conventional belief
...

Figure 17: Example of the collected dialogues in QuAC where the arrows define the continuation set of dialogue acts. Source: (Choi et al., 2018).

3 Developed System

In this third chapter the CQA dialogue system developed in this project will be explained. This system, named as **BERT fine-tuning for dialogue** by us, is a modification of the BERT (Devlin et al., 2018) fine-tuning approach for QA that will be able to predict dialogue acts together with the answer span of text.

3.1 BERT original system

BERT [8] is a language representation model that is designed for pre-training deep bidirectional representations jointly conditioned on right and left contexts. Once obtained the bidirectional language representations, the model can be fine-tuned for a wide range of NLP tasks with just one additional output layer. The authors have shown that BERT is able to outperform state-of-the-art systems without great task-specific architecture modifications.

3.1.1 Architecture

BERT model implements a stacked Transformer Encoder as the one presented in (Vaswani et al., 2017). The encoder of the Transformer (see Figure 18) is composed of two sub-layers: a multi-headed self-attention mechanism and a fully connected feed-forward network. Apart from that, residual connections (He et al., 2016) are employed around each of the two sub-layers, followed by layer normalization (Ba et al., 2016). So if we have an input representation x the output of each sub-layer would be the following: `LayerNorm(`x `+ Sublayer (`x`))`, where `Sublayer(`x`)` is the function implemented by the sub-layer itself.

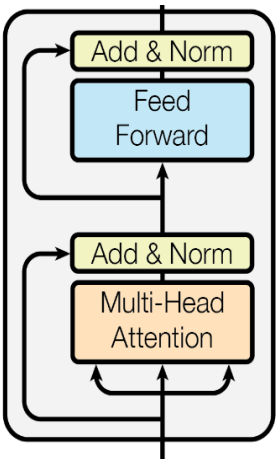

Figure 18: Architecture of the encoder of the Transformer. Source: (Vaswani et al., 2017).

[8] BERT model is available at `https://github.com/google-research/bert`

Multi-Head Self-Attention

Inside of this first sub-layer, the transformer employs a multi-headed self-attention mechanism. The self-attention function named by the authors as "Scaled Dot-Product Attention", takes three vectors as input: the query vector $q \in d_k$, the key vector $k \in d_k$ and the value vector $v \in d_v$, however, in practice the set of queries, keys and values are computed simultaneously packing the vectors together in matrices Q, K and V. Once having this Q, K and V matrices, attention is calculated with the following equation:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}}V) \tag{6}$$

This equation is identical to the dot-product attention, except for the scaling factor of $\frac{1}{\sqrt{d_k}}$. This factor is used for counteracting the effect of the $softmax$ function being pushed into regions with extremely small gradients. This usually happens when the dot products grow large in magnitude. A visual representation of it can be seen in the right side of Figure 19.

Moreover, instead of performing a single attention function, the multi-head attention performs h different Scaled Dot-Product functions for afterwards concatenating and projecting the d_v dimensional outputs as the final attended values. This can be seen in the left side of Figure 19. The motivation behind the amount of different heads is to extend the model's ability to focus on different positions of the text, giving to the model the capacity to resolve different phenomena as co-reference for example. Multi-head attention could be defined in the following way:

$$MultiHead(X) = Concat(head_1, ..., head_h)W^O$$

$$head_i = Attention(XW_i^Q, XW_i^K, XW_i^V)$$

Where, $X \in \mathbb{R}^{N \times d_model}$ is a matrix with the packed input embeddings $\{x_0, x_1, ..., x_n\}$ and $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$ and $W^O \in \mathbb{R}^{hd_v \times d_{model}}$ are the parameters. Here, d_{model} refers to the hidden dimension of the model and d_k and d_v are defined as $d_k = d_v = d_{model}/h$. Due to the reduced dimensionality of each of the heads, the total computation time of the multi-head attention is similar to that of single-head attention with full dimensionality.

Position-wise Feed-Forward Network

Apart from the attention sub-layer, a fully connected feed-forward network is also implemented. This network consist of two linear transformations with a ReLU activation:

$$FFN(x) = max(0, xW_1 + b_1)W_2 + b_2 \tag{7}$$

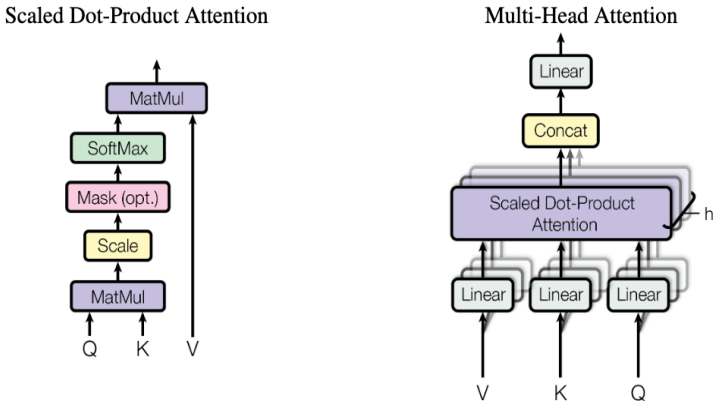

Figure 19: In the left the Scaled Dot-Product Attention can be appreciated. In the right the Multi-Head Attention that consists of several attention layers in parallel. Source: (Vaswani et al., 2017).

The parameters are not shared between layers and their input and output dimensionality is of d_{model}, however, the inner-layer has a size of $4d_{model}$.

3.1.2 Input

The input for this model consists of the summation of **token**, **segment** and **positional** embeddings for representing the cases of both, single and a pair of sentences, in an unambiguous way.

The input sentence **tokens** are represented with WordPiece (Wu et al., 2016) embeddings. Wordpiece embeddings are achieved by breaking words into word pieces given a word unit inventory. Special word boundaries are added before the training of the model, this way the original word sequence can be recovered from the Wordpiece sequence without ambiguity. An example of a word sequence and its corresponding Wordpiece sequence is given below:

Word tokenizer = `He is an unaffable man`

Word piece tokenizer = `He is an un ##aff ##able man`

The word unit inventory is learned from large amounts of text automatically and incrementally by running a greedy algorithm. This algorithm returns a user-specified number of word units that is chosen without any semantics notion. The algorithm goes as follows:

1. Initialize the word unit inventory with all the basic characters for a given language.
2. Build a language model on the training data with the inventory from 1.

3. Generate new word units by combining two units from the current word inventory. Choose the new word unit out of all possible ones that increases the likelihood on the data the most when added to the model.

4. Goto 2 until the greedily chosen number of word units is reached or the likelihood increase falls below a certain threshold.

Apart from that, two special **token embeddings** are also used: [CLS] and [SEP].

- The [CLS] token is appended as the first token of every sequence and its final hidden state is used as a special classification embedding.

- The [SEP] token is used to separate the two input sentences.

As for the learned **segment embeddings**, a sentence A embedding is added to every token on the first sentence and a sentence B embedding to every token on the second sentence. In the case of single-sentence inputs only A embeddings are used.

The learned **positional embeddings** support a sequence length up to 512 tokens, so the input total length will be limited to this amount of tokens. A visual representation of the input is given in Figure 20.

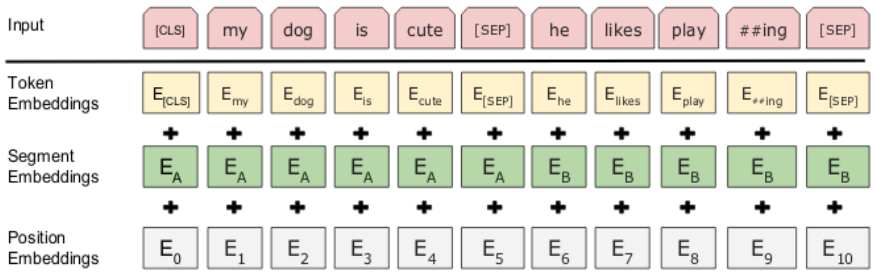

Figure 20: Input representation of the BERT model. The input embeddings are the sum of the token, segmentation and positional embeddings. Source: (Devlin et al., 2018).

3.1.3 Pre-training

For the pre-training of the bidirectional language representation model the authors propose two unsupervised prediction tasks: **masked LM** and **next sentence prediction**. The loss function used in the multitask pre-training is the sum of the mean masked LM likelihood and mean next sentence prediction likelihood.

Masked LM

In this prediction task a 15% of the input WordPiece tokens in each sequence are masked at random and then the system has to predict which was the original masked token. This straightforward approach enables the system to obtain a bidirectional representation of words, but using the [MASK] token the 100% of the time would end up on a mismatch between the pre-training and fine-tuning steps, since this token is not present in the fine-tuning process. In order to fight this issue the authors suggest the following:

- In a 80% of the cases use the actual [MASK] token. For example converting "Urretxu is so beautiful" into "Urretxu is so [MASK]".
- In a 10% of the cases replace the token with a random word. For example converting "Urretxu is so beautiful" into "Urretxu is so computer".
- In a 10% of the cases do not change the word.

The motivation behind this distribution of percentages is that using only random words would teach the system that the observed word is never correct and, on the contrary, keeping always the original word, the system would just trivially copy the non-contextual embedding. The use of these three masking strategies ends up on the Transformer encoder not knowing which words it will be asked to predict, so it is forced to keep the contextual representation of every input token. In Figure 22 a visual representation of the task can be appreciated. Here, the hidden representation of the last stacked transformer encoder is fed into a classification layer consisting of a fully-connected layer with GELU (Hendrycks and Gimpel, 2016) activations and layer normalization. Afterwards this output is multiplied by the embedding matrix for transforming it to vocabulary dimension and the probability of each word is calculated by a softmax. As for the loss, the masked words will be the only ones taken into account.

Next Sentence Prediction

In many NLP tasks as QA for example, understanding sentence relationships is something essential. Language models do not easily capture these relationships so with the Next Sentence Prediction task the model is expected to understand sentence relationships in a clearer way. The training of this task is carried out by choosing two sentences A and B from the training corpus, where B is the actual next sentence for A only in a 50% of the cases, in the rest of the cases B is just a random sentence from the corpus. For example:

Input = `[CLS] I live in [MASK] [SEP] Urretxu is so [MASK] [SEP]`
Label = `IsNext`

Input = `[CLS] [MASK] live in Zumarraga [SEP] I [MASK] trains [SEP]`

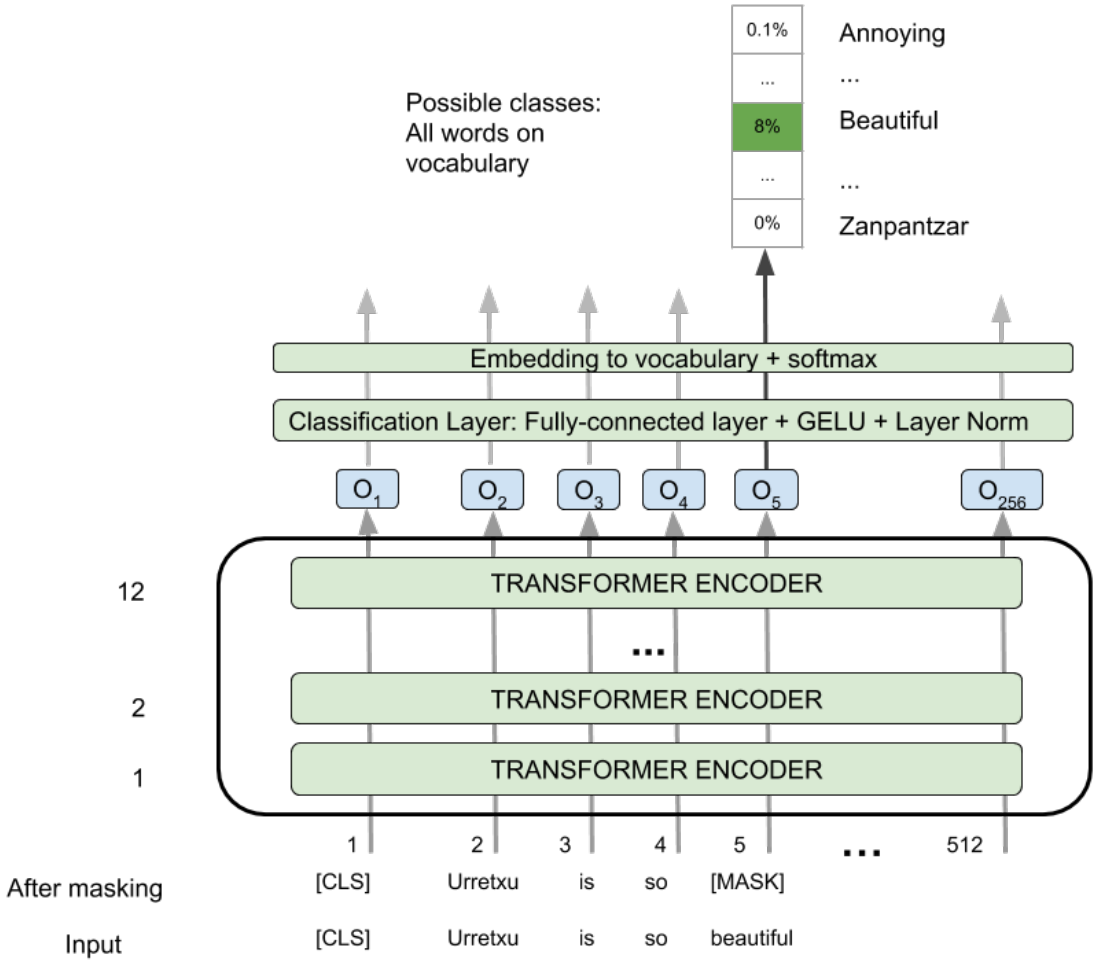

Figure 21: Visual representation of the Masked LM pre-training task.

Label = `NotNext`

A visualization of the pre-training task can be seen in Figure 22. Here, a feed forward neural network with parameters $N \in \mathbb{R}^{2 \times d_{model}}$ is added for classification after the last hidden layer of the [CLS] token.

3.1.4 Fine-tuning for QA

The authors of BERT present a wide range of fine-tuning approaches for different NLP tasks, however, we will just focus on the QA system, which is tested using the SQuAD dataset presented in section 2.2.3. So, for this fine-tuning approach, given a question and a passage containing the answer, the system will have to predict the answer text span in this passage. This task is similar to the one we want to overcome, where apart from the

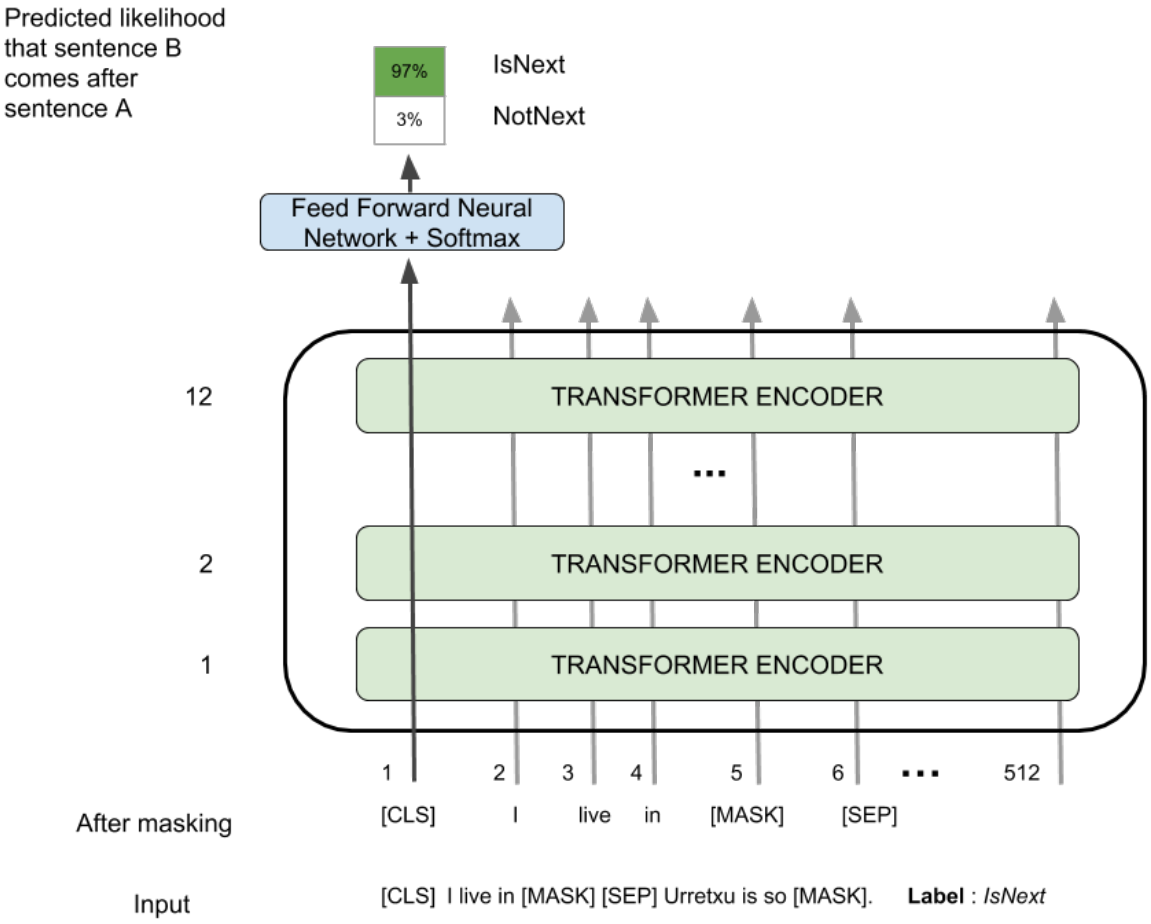

Figure 22: Visual representation of the Next-Sentence Prediction pre-training task.

answer text span we will have to predict some dialogue acts. With the aim of predicting the dialogue acts presented in QuAC, we will modify the original system to suit our interests.

The input representation is done by merging the question and passage into one sequence. Then the question and passage are separated by using the A embeddings for the question and the B embeddings for the passage. In the cases were the input sequence length is larger than 512 tokens, a sliding window is used for feeding the data to the network and then, the span with the largest context is greedily chosen. Even if in the fine-tuning step all the parameters are updated, there are only two new vectors to be learned for answer span spotting: a start vector $S \in \mathbb{R}^d_{model}$ and an end vector $E \in \mathbb{R}^d_{model}$, being d_{model} the size of the hidden dimension. The probability of each token to be the start token is computed by the following softmax:

$$P_i = \frac{e^{S \cdot T_i}}{\sum_j e^{S \cdot T_j}} \tag{8}$$

where, $T_i \in \mathbb{R}^d_{model}$ denotes the final hidden vector from BERT for the i^{th} input token. A visualization of this approach can be seen in Figure 23. The same formula is used for the end token and the maximum scoring span is taken as prediction.

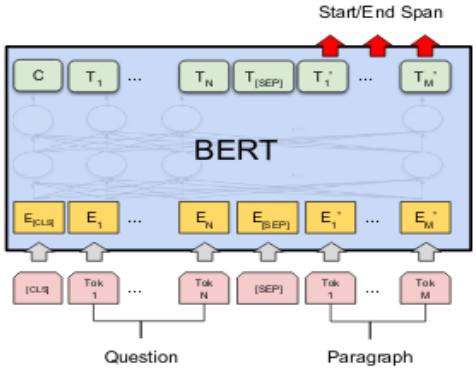

Figure 23: SQuAD fine-tuning approach of the BERT model. Source: (Devlin et al., 2018)

3.1.5 Available pre-trained models

Two BERT models are made available [9] by the authors:

- **$\text{BERT}_{\text{BASE}}$:** with 12 transformer encoder layers, a hidden size of $d_{model} = 768$ and $h = 12$ self-attention heads. The total amount of parameters in this model is of 110M parameters.

- **$\text{BERT}_{\text{LARGE}}$:** with 24 transformer encoder layers, a hidden size of $d_{model} = 1024$ and $h = 16$ self-attention heads. The total amount of parameters in this model is of 340M parameters.

In all the results we report in this project the $\text{BERT}_{\text{BASE}}$ model is used since the vast amount of parameters included in the $\text{BERT}_{\text{LARGE}}$ model end up on out-of-memory issues in GPUs (Titan XP on our case). The developers of the original paper used Cloud Tensor Processing Units (TPUs) for pre-training BERT models and each of them took 4 days to complete. A visual representation of the two models can be seen in Figure 24.

[9] https://github.com/google-research/bert

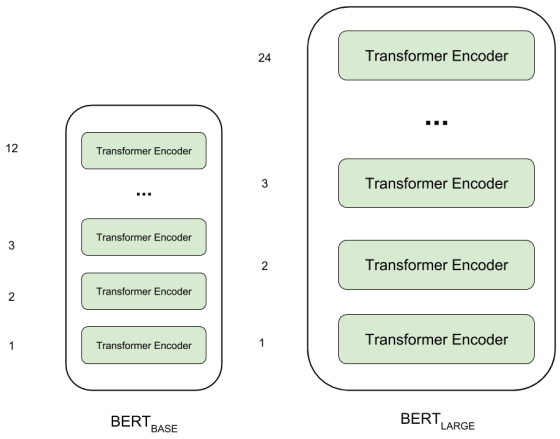

Figure 24: $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ models with 12 and 24 stacked encoders respectively.

3.2 BERT fine tuning for dialogue

Taking the BERT fine tuning approach for QA as an starting point, we have developed a fine tuning approach for CQA based dialogue that solves the following task:

Given a document or passage p of M tokens $\{p_1, p_2, ..., p_m\}$, a history of user questions and answers $Q_1, A_1, Q_2, A_2, ..., Q_{k-1}, A_{k-1}$ and a dialogue act set d of L discrete statements $\{d_1, d_2, ..., d_l\}$, we will have to find the answer A_k to question Q_k that will consist of the starting index i and ending index j of the answer on the passage together with a dialogue act list v.

In order to find the answer A_k we have used the same procedure of the previously mentioned QA fine-tuning approach. However, with the intention of predicting the list v of dialogue acts, we will modify the model for taking advantage of the final hidden vector of the [CLS] embedding. As mentioned in section 2.3.3 there are three different dialogue act sets presented in QuAC:

- Continuation, that contains: `follow up, maybe follow up or don't follow up`. So, in order to predict one of them, we will add a classification layer $W \in \mathbb{R}^{K \times d_{model}}$ to the last hidden vector of the [CLS] token followed by a softmax, where K is the number of classifier labels and d_{model} the hidden size.

- Affirmation, that contains: `yes, no or neither`. In the same way as before, we add another classification layer with the same dimensions for predicting affirmation dialogue acts.

- Answerability, that contains: `answerable or no answer`. In this case we use a different procedure for predicting this dialogue act set. Instead of adding a classification layer, we add the string `CANNOTANSWER` at the end of the passage. So, if the question has not an answer on the passage, the predicted span should be `CANNOTANSWER`.

A visual representation of the fine-tuning approach for dialogue can be seen in Figure 25.

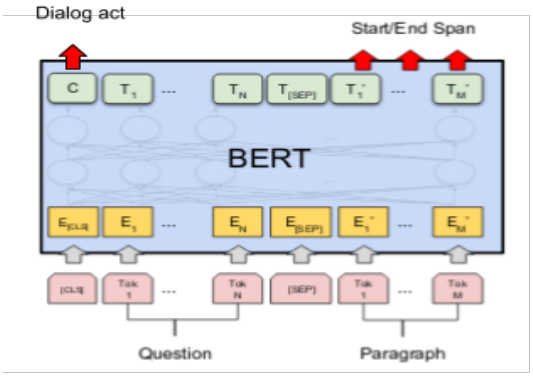

Figure 25: Modified BERT fine-tuning approach for dialogue act prediction. Source of original: (Devlin et al., 2018)

3.3 Experiments

We tested our system in the QuAC dataset that has been described in section 2.3.3. The parameters used for the pre-training and the fine-tunning can be seen in Table 5. All these parameters are the same of the original paper. The authors carried the pre-trainnig in 4 cloud TPUs and we fine-tunned the model on a GPU (Titan Xp). Due to the usage of different hardware, the batch size and epochs we used were smaller than the originally proposed ones, in order to avoid out-of-memory issues.

Parameter	Pre-trainning	Fine-tunning
Batch size	256	12(32)
Epochs	40	2(3)
Optimizer	Adam	Adam
Learning rate	1e-4	5e-5
Dropout	0.1	0.1
Training time	4 days	3 hours

Table 5: The hyper-paramaters of the BERT pre-training and fine-tunning. Between brackets, the parameters proposed on the original paper.

After having explained how the training of the model was done we will introduce the achieved results on the QuAC dataset. These results are obtained with the metrics suggested in the original paper:

- A **word-level F1**. Precision and recall are computed by considering the portion of words in the prediction and references that overlap after removing stop-words. In

this case two F1 values are reported: F1 and F1(All). In the computation of F1(All), all the questions are taken into account but in F1, questions with low human F1 (lower than 40) are removed. This threshold is chosen because a manual evaluation revealed a lot of low quality questions below it.

- **Human Equivalence Score**. HEQ is a new metric proposed for judging if the output of the system is as good as the human one. Two variants of HEQ are computed: (1) **HEQ-Q** that measures the percentage of questions for which system F1 exceeds human one, and (2) **HEQ-D** that measures the percentage of dialogues for which system F1 exceeds human one.
- For dialog acts, **accuracy** is reported.

	F1	HEQ-Q	HEQ-D	Yes/No	Follow up	F1(All)
BIDAF++(w/ 1-ctx)	59.9	54.9	4.7	86.5	61.3	57.5
BIDAF++(w/ 2-ctx)	**60.6**	**55.7**	**5.3**	86.6	**61.6**	58.3
BIDAF++(w/ 3-ctx)	**60.6**	55.6	5.0	86.1	**61.6**	58.1
Human performance	80.8	100	100	89.4	-	74.6

Table 6: Results obtained taking the context into account in the development set of the QuAC dataset as the test set is not publicly available.

	F1	HEQ-Q	HEQ-D	Yes/No	Follow up	F1(All)
BIDAF++(no ctx)	51.8	45.3	2.0	86.4	59.7	50.1
BERT (QA)	**53.2**	47.8	3.2	-	-	50.8
BERT (Dialogue)	51.7	46.4	3.0	**87.7**	61.1	49.1
Human performance	80.8	100	100	89.4	-	74.6

Table 7: Results obtained without taking the context into account in the development set of the QuAC dataset as the test set is not publicly available.

We have divided the obtained results into two tables: one with the results of the systems where the dialogue history is taken into account with the previous 1 (w/ 1-ctx), 2 (w/ 2-ctx) or 3 (w/ 3-ctx) answers (Table 6) and another one with the cases where the history is not considered at all (Table 7). The systems that appear on these tables are BIDAF++ explained in section 2.3.2, BERT QA explained in section 3.1.4 and BERT Dialogue explained in section 3.2. On the one hand, if we compare the F1 score results between both tables, it is obvious that the systems that model the dialogue history outperform the non contextual ones by a large margin (7.4 F1 points) when choosing the right answer span. However, for dialogue act prediction, non contextual systems perform really close even achieving the best results at Yes/No prediction.

On the other hand, if we just focus on the systems that do not model the dialogue history we can see how the BERT architecture outperforms the BIDAF++ one in all the cases,

suggesting that taking the dialogue history into account could end up on BERT model outperforming the BIDAF++(w/ 3-ctx) model. Nevertheless, it is worth mentioning the problem that BERT faces with long input sequences as its total input length is limited to 512 tokens and it can be hardly lengthened due to its feed forward nature. This mentioned limit in the input makes very hard the modelling of the history in this model.

To finish, we can also spot a difference between the results of the two BERT fine-tuning approaches, that has proven in quite an unexpected way that the dialogue act prediction is not helpful for the answer span selection. In this case, we end up getting worse F1 results in the dialogue fine-tunning approach than in the QA one.

4 Developed Dataset

In this section, the developed Community Question Answering (CQA) dataset will be presented. The main objective behind the development of this dataset is to realize if it is possible to mine dialogue from CQA forums as `https://stackexchange.com/`. If we prove that it is possible to obtain meaningful dialogues from forums, large task specific dialogues could be obtained for a wide variety of tasks. This possibility of obtaining task-specific datasets could open a new path on neural task-specific dialogue systems, as lack of available data is one of the greatest problems they face.

For this first CQA dialogue dataset we will just focus on the cooking domain as it is a general interest domain. So our initial idea is to obtain dialogue from a cooking question posted in a Stackexchange thread [10]. This idea is motivated by the type of answers that people tend to give in CQA forums as they could contain some latent dialogue. Answers in this type of websites are usually well argued and explained, suggesting that one of these answers could be splitted for replying more than one question. A Stackexchange thread is divided in the following way:

- The main **question**, **title** or **topic** of the thread. This consists of the initial question posted in the forum, the one that the answerer have to try to respond. Other users of the platform are given the opportunity to rate the formulated questions in order to quantify the question's quality. In the example of Figure 26 for example, the initial question is "How important is fresh ground coffee vs a good coffe grinder?" and has a rating of 8 points.

- The **background knowledge of the question**. When posting a question on Stackexchange, the questioner has the chance to give background information about his question in order to explain it for potential answerers. This text always appears behind the thread title and in Figure 26 is defined by "Given a choice between using a good coffee grinder a few days..."

- The **answers** given by the answerers of the forum. These pieces of text try to give an answer to the original question in a clear an extended way as other users can rate them and good feedback enables gaining reputation inside the platform. In the example of Figure 26 6 answers are given to the original question and the top rated has been given 7 positive votes by other users. At any moment from the first answer, the person who posted the original question can mark one of the answers as the accepted one, in order to show which one was the most useful for him.

In order to develop our idea two main steps have been followed: the Stackexchange cooking thread selection and the dataset collection. After carrying out these two steps we will end up with a dataset containing dialogue in which each of the questions have an answer and the passage where the answer was taken.

[10] `https://cooking.stackexchange.com/questions/36397`

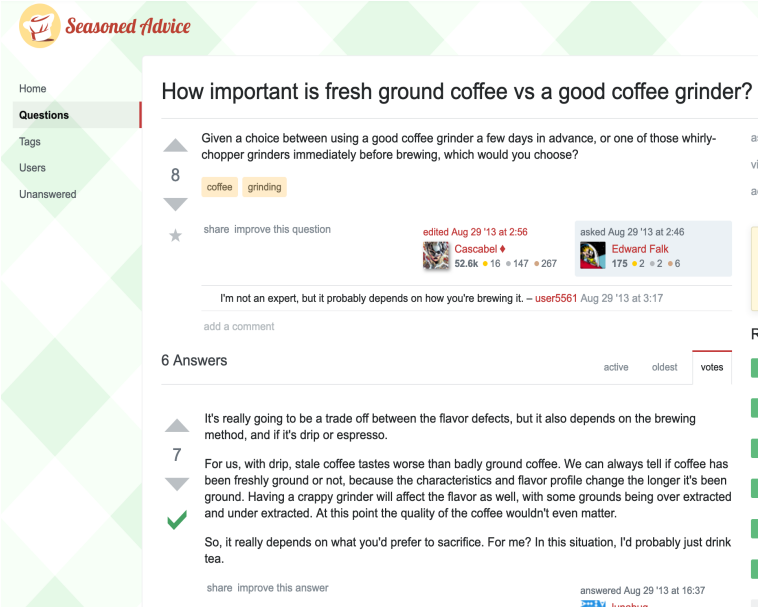

Figure 26: Example of a thread from the Stackexchange cooking community.

4.1 Cooking thread selection

Before starting with the dataset collection, question-answer pairs have to be mined from the Stackexchange data dump. We downloaded the data dump from September 2018 that includes 19818 threads in total. Once we have all the threads from the forum, we removed the threads with unaccepted answers and obtain 10079 out of the original 19818.

At this point, we did a small analysis of the scores of the questions and lengths of the answers. Attending to the scores, we realized that all the questions were scored in a range from $[-6, 240]$. After analyzing some random samples by hand, we deducted that even low scoring questions had a good quality, except for the ones with a negative score. Regarding the length of the answers, we saw that some of the answers were too long (maximum of 2960 words) for our task as very long passages could make our task very tedious. Taking all this into account, we applied some filters to these question-answer pairs:

- Questions with score <= 0 are removed, as we are not interested in badly asked questions.
- Question titles with more than one question mark are removed. The reason behind this filter is that we are interested in having the question titles as the first question of our dialogues and we are not interested in having more than one question per dialogue turn.
- The length of the answer has to be greater than 50 and shorter than 250 tokens. This way, we try to ensure that the answer passage is long enough for collecting dialogue, but not too long for avoiding tedious answer spotting.

- Answers that contain some html tags like hyperlinks, images, code... are removed.

After applying these filters we ended up with 3616 question-answer pairs, so we have 3616 different topics for collecting potential cooking dialogues.

4.2 Dataset collection

In order to perform the cooking CQA dialogue dataset collection we present an interactive task designed for two crowd-workers in Amazon Mechanical Turk. AMT is a crouwd-sourcing Internet marketplace that enables individuals (known as workers) and businesses (known as requesters) to coordinate themselves in order to use human intelligence on tasks that computers are nowadays unable to perform.

The workflow in this platform goes as follows: a requester posts an amount of Human Intelligent Tasks (HITs) that have to be completed. When posting one of this tasks, the requester has to specify a variety of options in order to carry on with it. First of all, the amount of money paid for each HIT has to be specified starting from $0.01. Then, the opportunity to filter the workers that have access to the posted HITs is given. This filtering can be done using general filters as: worker's minimal HIT approval rate, worker's mininal amount of approved HITs or worker's location. Apart from that, if the proposed HITs require complex previous understanding of the task, the requester can design a personal qualification task for his HITs and ask the workers to obtain a minimum mark on it if they want to perform the assignment. Once all these parameters are specified, the assignment is posted into AMT and the workers can have access to it. At this point, the workers that fulfil the previously specified requirements are able to choose if the proposed HITs are interesting for starting working on them. As soon as all the posted HITs are completed or the task is cancelled by the requester, he will have to accept or reject the achieved jobs and pay the workers with the previously defined money amount. Moreover, he will have the chance for paying them a bonus if he feels that the job has been done properly.

As it can be seen in this whole workflow, the workers are very unprotected in front of the requesters, so in order to fight this issue they have organized themselves in forums [11] where they give feedback of good and bad requesters and highlight the interesting HITs of the day.

4.2.1 Interactive Task

In our interactive task, we define a HIT as the task of generating a dialogue about cooking between two workers. These workers consist of a **curious** asking questions to a **cooking expert** about a certain topic from a Stackexchange cooking thread. These two roles are

[11] https://turkerview.com/ and `https://turkopticon.ucsd.edu/` for example

similar to the ones proposed in the QuAC task, however they have slight modifications.

The crowdworker that takes the curious role has access to the question posted in the forum together with the background information. Having this information, he must ask free text questions. The first question of every dialogue must be the one that appears in the title of the Stackexchange thread. An example of the questioner interface can be seen in Figure 27. On the right side of this interface the title of the cooking thread and its background knowledge is depicted. On the left side, a text box with the dialogue and an input box for free text question writing are given. Apart from that, send question and end chat buttons are also implemented for the curious role.

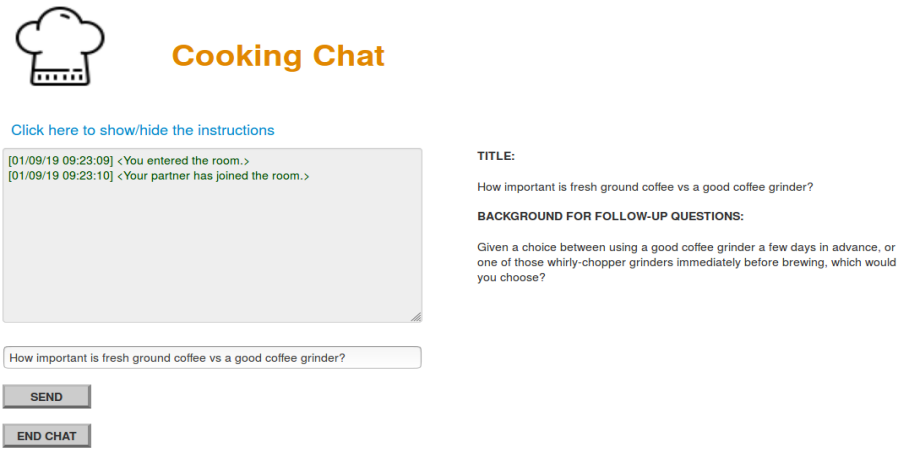

Figure 27: Interface of the curious role from the proposed dialogue collection interactive task.

In the case of the cooking expert, access to the answer is given and this worker must answer with a span of text selected from it. The cooking expert has the opportunity to modify the selected span of text in order to make the dialogue look more natural, but, we motivate minimal modifications copying the selected span of text directly to the answer. This strategy is very similar to the one used in the CoQA dataset, where they claim that 78% of the answers had at least one edit. But, as proven in (Yatskar, 2018) a 97.8 F1 upper-bound can be obtained by an extractive system, showing that the changes should be minimal in almost all the cases. Apart from the span of text, the expert has to give feedback with the same dialogue acts that are presented in QuAC, except for the *maybe follow up* act from the continuation set as we feel that it is not very intuitive:

- Continuation. It is used for leading the student to the most interesting topics: *follow up* or *don't follow up*.
- Affirmation. It is required when the question is a Yes/No question: *yes*, *no* or *neither*.

- Answerability. It will define if the question has an answer or not: *answerable* or *no answer*.

For the cases when the answer to a given question does not appear in the text the "I don't know" option has been also added. An example of the answerer interface can be seen in Figure 28. On the right side of the Figure interface for the selection of the span can be appreciated. On the left side, in a similar way to the curious interface, a text box with the dialogue is shown. However, in this case, we do not implement the end chat button as we do not want the answerer to end the chat.

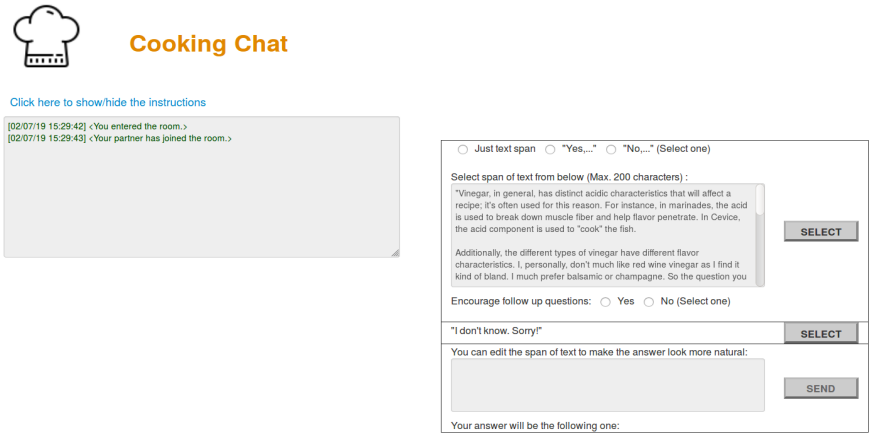

Figure 28: Interface of the cooking expert role from the proposed dialogue collection interactive task.

Finally, dialogues are ended when they reach a maximum of 8 question and answer pairs, when 3 unanswerable questions have been asked or when 10 minutes have passed in the collection task. The proposed limits are used for avoiding long and repetitive dialogues as the cooking threads are very focused on a certain topic and in many cases it is impossible to have non-repetitive long conversations.

The implementation of this interface has been done by modifying the source code of the Collaborative Communicating Agents (CoCoA) dialogue framework [12] in order to suit our interactive task characteristics.

4.3 Trials and tribulations with AMT

In order to carry out the actual collection of the dataset, we decided to perform an incremental amount of trials in AMT for tuning our interface and the filters that the platform offers to us. Still, even before of the incremental trials in AMT, we performed a small in-house test among the LIHLITH project participants for testing the correct performance

[12] `https://github.com/stanfordnlp/cocoa`

of the application. Once we realized we had a robust interface, we started with the AMT attempts.

1. In the first trial, we set up 25 question and answer pairs with the following worker requirements:

   - HIT approval rate $\geq$ %98.
   - Approved HITs $\geq$ 1000
   - Location of the workers: US and Canada
   - Workers had to be qualified as Masters. This master qualification is automatically given by Amazon taking the worker's performance into account. Workers that are highlighted as masters are supposed to be the greatest ones in the platform, however, they are also the most demanding ones and do not usually take part in very complex tasks, since receiving a reject in a certain tasks could end up on them loosing their master status.

   This first experiment was not very promising as almost no worker accessed our server.

2. In the second trial we decided to keep the 25 question and answer pairs of the first one as none of them was completed. Nevertheless, we decided to change the used worker requirements:

   - HIT approval rate $\geq$ %95.
   - Approved HITs $\geq$ 1000
   - Location of the workers: US and Canada
   - Workers are not longer required to be qualified as masters.

   With these softer requirements all the posted HITs were completed in almost two hours, but the quality of the achieved dialogues was very low. We only achieved 3 understandable dialogues out of 25. An example of the achieved dialogues can be seen in Figure 29.

3. Despite the low rate of valid dialogues, we observed that in the cases were the workers understood the task, the mined conversations were very sensible and close to our expectations. So, having this information, we thought that the best choice was to design a qualification test to test the worker's understandings of the task, asking questions about the instructions. We did a couple of trials with the same requirements as before and expecting a minimum of 4 out of 6 and 2 out of 3 correct answers in the test. In both of these trials we did not succeed on getting any dialogues as very few people was simultaneously connected to out server.

4. After all these four unsuccessful trials we returned to the initial requirements, but removing the master qualification requirement and adding more English speaking

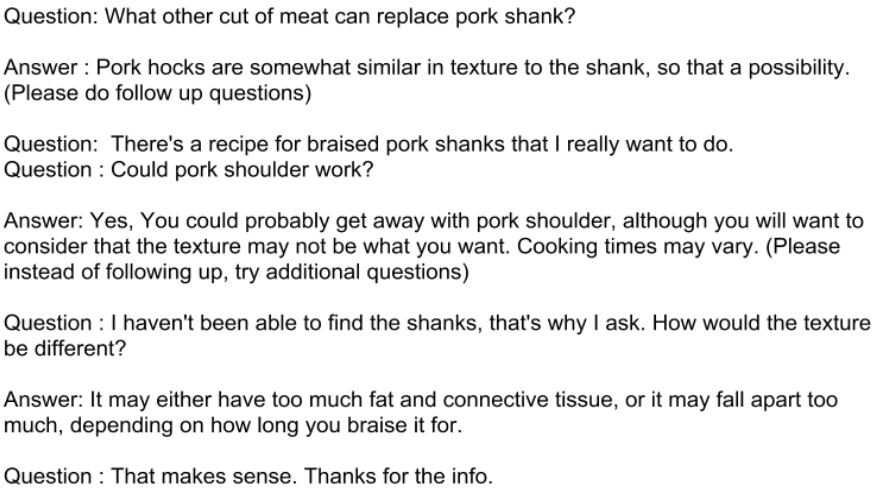

Figure 29: An example of the dialogues obtained in the second AMT trial of our interactive task for mining cooking dialogues.

country locations. Apart from that, we did a small change in the interface, only allowing the workers to end the task when the developed dialogue had at least 2 turns and contained at least one span selected answer. This way we avoided having dialogues containing only "I don't know sorry" answers. With these changes our requirements ended up as follows:

- HIT approval rate $\geq$ %98.
- Approved HITs $\geq$ 1000
- Location of the workers: US, Canada, UK, Australia.
- A minimum length of 2 question and answer pairs for the dialogues (Controlled by our interface).
- At least having one answer that is not "I don't know sorry".

With this new requirements we achieved to get 24 out of 24 meaningful dialogues. These dialogues can be appreciated in Appendix A.

4.4 Data collection and analysis

After carrying out all the mentioned trials, we decided to run the actual crowdsourcing of the dataset of 400 dialogues with the following specifications:

- HIT approval rate $\geq$ %98.
- Approved HITs $\geq$ 1000

- Location of the workers: English speaking countries.
- A minimum length of 2 question and answer pairs for the dialogues (Controlled by our interface).
- At least having one answer that is not "I don't know sorry".
- A price of $0.10 for doing the HIT and a bonus of $0.33 for each question or answer given during the task except for the "I don't know sorry" case where $0.05 is paid. This difference in the pays is used for motivating the workers to force themselves to find the actual answer in the passage, because answering "I don't know" is less demanding than searching for the correct answer span. The average price for each dialogue is of $3.2.

After 24 hours we stopped the task as we managed to get 390 dialogues. With the intention of doing a small preliminary analysis of the obtained dialogues some statistics have been computed and can be appreciated in Table 8. This analysis is just preliminary as the dataset has been recently obtained. Apart from the computed statistics, and with the aim of having a brief overview of the quality of the dialogues, each of the workers was asked to give some feedback after completing a HIT. In the case of the questioner he had to measure his satisfaction from 1 to 5 and in the case of the answerer he had to measure the sensibility and helpfulness of his partner from 1 to 5. As it can be seen in the Table 8 the obtained results are quite good ≥ 3.9 in all the cases, so we can think that the obtained dialogues are meaningful even if a deeper analysis would be required before the usage of the dataset.

If we pay attention to the results of Table 8 we can see how our dataset is quite small, specially when compared to QuAC and CoQA. However, by analyzing the dialogues we can conclude that the followed strategy for domain specific dialogue dataset crowdsourcing is sound,as shown by the fact that the average tokens per questions and answers are closer to real dialogues compared to the other datasets. Analysing the low average of tokens per answer in CoQA suggests that this dataset is closer to QA than dialogue as human dialogues tend to be longer and argued, not just single answers. Apart from that, our developed dataset has the lower ratio of questions per dialogue, although this was quite expected as we are working with very specific domains and long dialogues are not required in this cases.

As for the dialogue acts distribution, the dataset shows a very similar tendency to QuAC dataset as we use a almost the same dialogue act set. Here CoQA suffers from the lack of dialogue acts and ends up on having almost all of its questions answearable, facing the same issues as SQuAD 1.0 (see section 2.2.3). Last but not least, we should mention the distribution of extractive and abstractive answers that is similar to the one in CoQA, suggesting that in a similar way to (Yatskar, 2018) we could develop both robust extractive and abstractive systems for our developed dataset.

Dataset	**Ours**	**QuAC**	**CoQA**
Questions	1,808	98,407	**127,000**
Dialogues	390	**13,594**	8,399
Tokens/Question	**10.24**	6.5	5.5
Tokens/Answer	**17.63**	14.6	2.7
Questions/Dialogue	4.63	7.2	**15.2**
Extractive %	72.65	**100**	66.8
Abstractive %	27.35	0	**33.2**
Yes/No %	22.27	**25.8**	-
I don't know %	**31.17**	20.3	1.3
Satisfaction of answerer	3.9/5	-	-
Sensibility of answerer	4.27/5	-	-
Helpfulness of answerer	4.10/5	-	-

Table 8: Statistics of the developed dataset compared with the QuAC and CoQA dataset.

5 Conclusion and future work

In this thesis project we have analysed current state-of-the-art QA and dialogue systems and datasets. This analysis has guided us on our path to develop an end-to-end dialogue system. We realized that QuAC is the most interesting dataset for QA based dialogue due to its size and the implementation of the dialogue acts they propose, which gives QA dialogue datasets the notion of something that was previously not implemented on them and that is crucial for carrying out a good dialogue (see section 2.3.1). The adaptation of the BIDAF++ system, traditionally used for QA, for this dataset has shown to achieve great results, and therefore we replicated them in the **QuAC** dataset.

Apart from replicating the results of this state-of-the-art system, we adapted the QA fine-tuning approach of the **BERT** model and improved the state-of-the-art results when the dialogue history is not considered. However, we realized that the feed forward network nature of the BERT model hinders the management of input sequences that are longer than 512 tokens, making the modelling of the history in this system very challenging.

Regarding data collection, we created a cooking domain dialogue dataset using the Amazon Mechanical Turk crowdsourcing platform. The collection and the analysis of the 390 achieved dialogues and 1,808 QA pairs has been done with encouraging results for continuing with further data collection. This opens a new path on CQA dialogue systems, as new meaningful datasets are the best way of feeding future research. It is worth mentioning that the importance of building proper datasets and evaluation procedures has been lately highlighted by the research community as we can appreciate in Mikel Artetxe's words from the well-known Sebastian Ruder's NLP blog [13]:

"Perhaps the biggest problem is to properly define the problems themselves. And by properly defining a problem, I mean building datasets and evaluation procedures that are appropriate to measure our progress towards concrete goals."

For the future I would like to develop the following work as a PhD candidate:

- Using the BERT model for extracting contextual embeddings and plugging them in BIDAF++ system. This is motivated by the difficulty that BERT faces with sequences longer than 512 tokens. This issue, suggested us that maybe the RNNs are the method to go for the development of end-to-end dialogue systems.
- Modelling the dialogue history in BIDAF++ with a more complex approach than simple concatenation as in the BIDAF++ for dialogue system. The most appealing approach is using a hierarchical attention similar to (Miculicich et al., 2018), that applies this hierarchy for document level machine translation.

[13] `http://ruder.io/4-biggest-open-problems-in-nlp/`

- Doing a further analysis on the collected dialogues and cleaning the data for making out of domain experiments of dialogue systems to extend the 390 potential dialogues.
- Collecting extra 1600 dialogues with the funding of the LIHLITH project for building a bigger dataset with the intention of making it available for the research community.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Ricardo Baeza-Yates, Berthier de Araújo Neto Ribeiro, et al. *Modern information retrieval*. New York: ACM Press; Harlow, England: Addison-Wesley,, 2011.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014a.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014b.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. Quac: Question answering in context. *arXiv preprint arXiv:1808.07036*, 2018.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Dan Hendrycks and Kevin Gimpel. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *arXiv preprint arXiv:1606.08415*, 2016.

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

Dan Jurafsky and James H Martin. *Speech and language processing*, volume 3. Pearson London, 2014.

Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

Lesly Miculicich, Dhananjay Ram, Nikolaos Pappas, and James Henderson. Document-level neural machine translation with hierarchical attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2947–2954. Association for Computational Linguistics, 2018. URL `http://aclweb.org/anthology/D18-1325`.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Amit Mishra and Sanjay Kumar Jain. A survey on question answering systems with classification. *Journal of King Saud University-Computer and Information Sciences*, 28 (3):345–361, 2016.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

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

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

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

Siva Reddy, Danqi Chen, and Christopher D Manning. Coqa: A conversational question answering challenge. *arXiv preprint arXiv:1808.07042*, 2018.

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

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

AM Turing. Mind. *Mind*, 59(236):433–460, 1950.

Ronald D Vale. The value of asking questions. *Molecular biology of the cell*, 24(6):680–682, 2013.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

Mark Yatskar. A qualitative comparison of coqa, squad 2.0 and quac. *arXiv preprint arXiv:1809.10735*, 2018.

Tao Yu, Michihiro Yasunaga, Kai Yang, Rui Zhang, Dongxu Wang, Zifan Li, and Dragomir Radev. Syntaxsqlnet: Syntax tree networks for complex and cross-domaintext-to-sql task. *arXiv preprint arXiv:1810.05237*, 2018a.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. *arXiv preprint arXiv:1809.08887*, 2018b.

A Collected Dialogues

Dialogue 1

Question: Smoked a chicken and the skin came out like boot leather! What can I do?
Answer : protect the skin during the long slow smoking process. You may want to pull the cheesecloth off about 30 minutes before you are done.(Please instead of following up, try additional questions)
Question: Is 4 hours too long to keep in the smoker?
Answer : I don't know. Sorry!
Question: What about using a little apple juice and vegetable oil on the skin?
Answer : yes you can(Please instead of following up, try additional questions)
Question: What is a good temperature for the smoker?
Answer : 400 degrees(Please instead of following up, try additional questions)
Question: Do you have any other tips to prevent this from happening again?

Dialogue 2

Question: Does poking steaks and other meats cause flavor loss?
Answer : Yes, "This is totally fine. Meat gets poked and prodded all the time, whether to test it, to pick it up with tongs, or sticking a fork in it to cut. Meat just is not that fragile. (Please do follow up questions)
Question: Is there a way I can alter the flavor on the meat?
Answer : I don't know. Sorry!
Question: How should I reduce the "dead taste"?
Answer : Yes, Things like smashing with a tenderizer (Please do follow up questions)
Question: Great, I will give that a try on my meat. A friend said that I'm ruining the steak and that it causes flavor and juice losses. Is this the case?
Answer : Meat just is not that fragile.(Please do follow up questions)
Question: Okay, how should I place the meat on the grill?
Answer : I don't know. Sorry!
Question: Okay thanks!

Dialogue 3

Question: Ginger starting to sprout - can I still use it?
Answer : Yes, I find that when sprouted, the ginger just loses a bit of flavour, that's all. (Please do follow up questions)
Question: Do I have to prepare it in a different way than non sprouted ginger?
Answer : I don't know. Sorry!
Question: Does it have a different taste than non sprouted ginger?

Answer : Yes, the ginger just loses a bit of flavour, that's all. If you let it grow for a very long time and it becomes all shriveled, then you are talking about a flavourless piece of ginger. (Please do follow up questions)
Question: What about the sprouts, are they edible?
Answer : I don't know. Sorry!
Question: Can I plant sprouted ginger to make more ginger roots?
Answer : I don't know. Sorry!

Dialogue 4

Question: In general, can I substitute egg species for either egg-only or egg-centric dishes?
Answer : There may be some issues with substituting in dishes due to varying fat content of the yolks and such...(Please do follow up questions)
Question: Do you think that the smaller eggs like quail eggs would have more flavor for a quiche?
Answer : I don't know. Sorry!
Question: I think that duck eggs would make a great addition to chili rellenos because of the high fat content. Do you think that would be something great to have for breakfast?

Dialogue 5

Question: What is an arepa?
Answer : White pre-cooked corn flour, water, pinch of salt and cooked in the pan(Please instead of following up, try additional questions)
Question: Can you use anything besides water to cook them?
Answer : Yes, cooked in the pan and in the oven and decide for yourself which you prefer (Please do follow up questions)
Question: Is it better to cook them in the oven or in the pan?
Answer : I don't know. Sorry!

Dialogue 6

Question: Fluffy texture in a Spanish tortilla
Answer : "If you want it fluffy, you'll need air(Please do follow up questions)
Question: How can I add air to it?
Answer : The easiest way to accomplish this is to crack the eggs in a bowl, whip them (with a whisk or a fork) for a couple of minutes(Please do follow up questions)
Question: Do I add it to a hot pan?
Answer : I don't know. Sorry!
Question: When do I put it under the grill?
Answer : I don't know. Sorry!
Question: Do I add spices?

Answer : I don't know. Sorry!

Dialogue 7

Question: Does poking steaks and other meats cause flavor loss?
Answer : Yes, "This is totally fine. Meat gets poked and prodded all the time, whether to test it, to pick it up with tongs, or sticking a fork in it to cut. Meat just is not that fragile. (Please do follow up questions)
Question: Is there anything I could do wrong to give the meat a 'dead taste'?
Answer : I don't know. Sorry!
Question: How much moisture will a steak lose whilst cooking?
Answer : Yes, Things like smashing with a tenderizer are much more violent and do affect the texture (that's kind of the point), so that's not a great comparison. But this is a really minor thing (Please do follow up questions)
Question: Would you not recommend using a tenderizer?
Answer : No, This is totally fine. (Please do follow up questions)
Question: how long should meat be tenderized for ?
Answer : I don't know. Sorry!

Dialogue 8

Question: Does poking steaks and other meats cause flavor loss?
Answer : This is totally fine. Meat gets poked and prodded all the time, whether to test it, to pick it up with tongs, or sticking a fork in it to cut. Meat just is not that fragile.(Please instead of following up, try additional questions)
Question: Would you recommend using a meat tenderizer?
Answer : Things like smashing with a tenderizer are much more violent and do affect the texture (that's kind of the point), so that's not a great comparison. But this is a really minor thing(Please instead of following up, try additional questions)
Question: Is it possible to bruise the steak using a tenderiser?
Answer : Things like smashing with a tenderizer are much more violent and do affect the texture (that's kind of the point), so that's not a great comparison. But this is a really minor thing.(Please do follow up questions)
Question: How does tenderising affect the flavour?
Answer : I don't know. Sorry!
Question: Would you suggest using a flavour enhancer?
Answer : I don't know. Sorry!

Dialogue 9

Question: How do I know when a chicken breast has cooked through?
Answer : Yes, A thermometer is the only way to be sure. (Please do follow up questions)

Question: Are there other ways to tell? In the event I did not have a thermometer for some reason.
Answer : Yes, you should able to learn the average cooking time, and outwards cues of color and texture that match the right internal temperature. (Please do follow up questions)
Question: What are some of the outward cues?
Answer : Internally, the meat should look opaque and white.(Please do follow up questions)
Question: Okay, how about externally?
Answer : I don't know. Sorry!
Question: What temperature should the meat be cooked at in order to cook all the way through?
Answer : I don't know. Sorry!
Question: Is cutting the meat open a good way to make sure it is cooked through?
Answer : No, For methods with a consistent level of heat (stove, oven), you should able to learn the average cooking time, and outwards cues of color and texture that match the right internal temperature. (Please do follow up questions)
Question: What should the internal temperature be exactly?
Answer : I don't know. Sorry!

Dialogue 10

Question: I smoked a chicken and the skin came out like boot leather, can you help me figure out what went wrong?
Answer : Yes, "Take cheesecloth and soak it in melted butter and drape it over the bird before you put it in the smoker. This will protect the skin during the long slow smoking proces (Please do follow up questions)
Question: Okay, thanks. I had the temperature at 225-250 for 4 hours. Are both of those find when I try this again in the future?
Answer : I don't know. Sorry!
Question: If I do not have a cheesecloth is there something else I could use?
Answer : melted butter (Please instead of following up, try additional questions)
Question: What it I do not have melted butter to soak the cheesecloth in? Is there a substitute?
Answer : I don't know. Sorry!
Question: How much melted butter should I use?
Answer : I don't know. Sorry!

Dialogue 11

Question: Filtered or non-filtered soymilk maker? Filtered or non-filtered soymilk maker, which one would you suggest buying?

Answer : Yes, I have an old SoyaQuick (mine has a filter, newer models don't), and I think it was largely a cleaning concern. (Please do follow up questions)
Question: Do you most people will prefer using a newer version of the soyamilk maker?
Answer : I don't know. Sorry!

Dialogue 12

Question: Is it possible to cook a meatloaf using clear glass Pyrex containers?
Answer : Yes, "I see no reason you couldn't use that Pyrex set for a meatloaf - I've used glass casseroles for meatloaf before (so glass in general is no problem), and that set says the bowls are oven safe. (Please do follow up questions)
Question: Great thanks! I've got a pyrex set...are they generally microwave safe?
Answer : I don't know. Sorry!
Question: No worries...do you know if they are oven safe?
Answer : Yes, the bowls are oven safe. (Please instead of following up, try additional questions)
Question: Ok cool. What was the result when you cooked meatloaf using pyrex...big hit, or total disaster?
Answer : Yes, I'd recommend wrapping it in aluminum foil to help it keep its shape. (Please instead of following up, try additional questions)
Question: Oh is that what you did?
Answer : Yes, As for the cookie sheet method, I would be afraid of it falling apart as you described, but if you were to go that route, I'd recommend wrapping it in aluminum foil to help it keep its shape. (Please do follow up questions)
Question: Cool...How large was the meatloaf you cooked?

Dialogue 13

Question: Ginger starting to sprout - can I still use it?
Answer : Yes, From a culinary perspective, I find that when sprouted, the ginger just loses a bit of flavour, that's all. (Please do follow up questions)
Question: Can I use it just like regular ginger, or is there something different I should do with it?
Answer : I don't know. Sorry!
Question: Does it require special preparation?
Answer : I don't know. Sorry!

Dialogue 14

Question: Is gelatin vegetarian?
Answer : Gelatin comes from a dead animal (unless they start harvesting it with arthroscopic probes :), so it is not a vegetarian ingredient. (Please do follow up questions)

Question: Is there a vegetarian alternative?
Answer : Yes, Yes, There are many other hydrocolloids, such as agar, that can be used to produce similar textures if needed. (Please instead of following up, try additional questions)
Question: So can you tell me what constitutes an ingredient as "vegetarian" or "vegan?"
Answer : I don't know. Sorry!
Question: That's ok. I was wondering if you had any recipes that include vegetarian gelatin?
Answer : I don't know. Sorry!

Dialogue 15

Question: What features should I look for when buying an espresso maker?
Answer : since it is the high-pressure components that are key to how they work, it is the quality and durability of these that tends to set the price point.(Please do follow up questions)
Question: Would a $30 machine have high pressure components like you mentioned?
Answer : I don't know. Sorry!
Question: Do you know what the differences between a $30 machine and a $500 machine are?
Answer : Yes, you generally get what you pay for with espresso machines. The more expensive domestic ones usually really do last much longer. (Please do follow up questions)
Question: What specific features should I look for?
Answer : My advice is to visit a couple of retailers that have staff dedicated to selling espresso machines.(Please do follow up questions)
Question: Will they be able to answer questions about different priced espresso machines?
Answer : Yes, they will also recognise when someone needs a domestic unit designed for daily use as opposed to one just for special occasions. (Please do follow up questions)

Dialogue 16

Question: Can I use garlic leaf for cooking?
Answer : Yes, When we have had garlic in our garden I have used the garlic leaves (Please do follow up questions)
Question: Can I dry the garlic leaves?
Answer : Regarding drying them, I have never tried it. Off the top of my head I can't think of any reason not to dry them for later use(Please do follow up questions)
Question: Would I be able to use the dried leaves like I use other herbs in cooking?
Answer : I don't know. Sorry!
Question: Do the leaves taste just like garlic?
Answer : They do have a garlicky flavor but are milder than garlic cloves(Please do follow up questions)
Question: Are they safe to eat?

Answer : Yes, tend to use them more as I would chives or garlic chives as in addition to having the milder flavor than the cloves they make for a quite nice presentation (Please do follow up questions)

Dialogue 17

Question: How long can you keep chocolate, and what is the best way to store it?
Answer : Regardless of type, all chocolate should be stored in a cool and low humidity (dry) place away from direct sunlight(Please do follow up questions)
Question: How long does chocolate last before losing flavor?
Answer : Dark chocolate will last for years. Milk and white chocolate will last for a much shorter time (a few months), because of their milk content.(Please do follow up questions)
Question: Once it gets that white stuff on the outside, is it done?
Answer : No, This kind of chocolate is still suitable for any application where the chocolate will be fully melted (most baking) (Please do follow up questions)
Question: What's the best way to store it for as long as possible?
Answer : all chocolate should be stored in a cool and low humidity (dry) place away from direct sunlight. It would be best to seal it in an air-tight container(Please do follow up questions)
Question: How long can I keep milk chocolate if properly stored?
Answer : Dark chocolate will last for years. Milk and white chocolate will last for a much shorter time (a few months), because of their milk content.(Please instead of following up, try additional questions)
Question: How do I know if chocolate has gone bad?
Answer : Improperly stored chocolate will develop bloom, which shows as a white or grey streaking or spotting on the surface.(Please do follow up questions)
Question: Will I get sick if I eat chocolate that has developed bloom?
Answer : No, This kind of chocolate is still suitable for any application where the chocolate will be fully melted (Please do follow up questions)

Dialogue 18

Question: What is the best way to store and manage tahini?
Answer : just spend some time and elbow grease to mix it back together again.(Please do follow up questions)
Question: What do I do when it separates?
Answer : When that happens, just spend some time and elbow grease to mix it back together again.(Please instead of following up, try additional questions)
Question: How do I recover it?
Answer : I don't know. Sorry!
Question: What are alternatives besides food processors?
Answer : I don't know. Sorry!

Question: What is the best way to store longterm?
Answer : if the tahini was stored in the fridge, it might take longer because everything will be harde(Please do follow up questions)
Question: Is there a better way?
Answer : I don't know. Sorry!

Dialogue 19

Question: How to cook good "arepas"?
Answer : I strongly recommend you experiment with the different flours, milk or water or half and half,(Please do follow up questions)
Question: If you wanted to sub corn flour would you just do it measure for measure? Sorry I meat white wheat flour
Answer : cookery is a living and evolving subject and very much a matter of personal taste. (Please do follow up questions)
Question: do you use water or milk?
Answer : water,(Please do follow up questions)
Question: and which is better oven or pan?
Answer : cooked in the pan is the traditional way(Please instead of following up, try additional questions)
Question: are these a sweet or savory food?
Answer : White pre-cooked corn flour, water, pinch of salt(Please do follow up questions)

Dialogue 20

Question: How good a substitute is callaloo for spinach?
Answer : d amaranth, so apologies if that is distinct from the type you have access to. As I recall, spinach is a bit sweeter and the leaves are a bit softer so they break down more readily(Please do follow up questions)
Question: Have you used both spinach and amaranth in recipes?
Answer : I've only had red amaranth,(Please do follow up questions)
Question: Do you have any recipes that you would recommend for red amaranth?
Answer : I don't know. Sorry!
Question: Would you recommend using amaranth as a replacement if someone cannot get spinach?
Answer : Yes, I've yet to find a recipe so touchy that one leafy green can't be substituted for another. (Please do follow up questions)
Question: Do you have a favorite leafy green recipe?
Answer : I don't know. Sorry!
Question: OK, thank you

Dialogue 21

Question: Why does this work? (defrosting steak)
Answer : he reason is, solids and liquids transfer heat better than gasses do. (Please do follow up questions)
Question: would you recommend this over a microwave?
Answer : Yes, Sandwich the steak between two pots, one of which has a large mass of warm water in it; the heat from the water will flow into the meat. (Please do follow up questions)
Question: How long would you expect it to take for a medium sized steak ?
Answer : I don't know. Sorry!
Question: Is there any damage to the steak with this method?
Answer : No, The reason is, solids and liquids transfer heat better than gasses do. (Please do follow up questions)
Question: Does the steak start to brown slightly?
Answer : I don't know. Sorry!

Dialogue 22

Question: Can I use garlic leaf for cooking?
Answer : Yes, They do have a garlicky flavor but are milder than garlic cloves. (Please do follow up questions)
Question: Is it safe to use?
Answer : I don't know. Sorry!
Question: Should I dry them?
Answer : Regarding drying them, I have never tried it. Off the top of my head I can't think of any reason not to dry them for later use but there may be issues that I just don't know about.(Please instead of following up, try additional questions)
Question: Do you know what the measurements would be compared to regular garlic?
Answer : I don't know. Sorry!
(04:47:10) **Question**: What types of food is garlic used for?
Answer : I don't know. Sorry!

Dialogue 23

Question: Why can't this ice cream scoop go in the dishwasher?
Answer : Yes, "I've accidentally run my scoop, a Zeroll with conductive fluid inside the handle, through the dishwasher. (Please do follow up questions)
Question: Will this ruin the scoop part of the ice cream scoop?
Answer : Yes, is that the fluid is meant to work at normal body temperature and when it gets too hot, like in a dishwasher, it solidifies (Please do follow up questions)
Question: what is the conductive fluid made of?

Answer : I don't know. Sorry!

Dialogue 24

Question: Why can't this ice cream scoop go in the dishwasher?
Answer : I believe what happened to mine (and what's happened to yours) is that the fluid is meant to work at normal body temperature and when it gets too hot, like in a dishwasher, it solidifies.(Please instead of following up, try additional questions)
Question: Does it do something to the metal?
Answer : I believe what happened to mine (and what's happened to yours) is that the fluid is meant to work at normal body temperature and when it gets too hot, like in a dishwasher, it solidifies.(Please instead of following up, try additional questions)
Question: Can I buy one that can go into the dishwasher?
Answer : I don't know. Sorry!
Question: If I was making india curry and don't have yogurt is there something I can substitute
Answer : I don't know. Sorry!
Question: If I wasn't going to used a ice cream spoon what else can I used to replace it
Answer : I don't know. Sorry!