One Million Agents Speaking All the Languages in the World

Miguel Ângelo António Ventura
Supervisor: Prof. Maria Luísa Torres Ribeiro Marques da Silva Coheur

Abstract

Currently, creating a conversational agent for a specific domain is an accessible task but the resulting agents have restricted knowledge due to the human effort needed to manually introduce the data.

Movie and TV shows subtitles are available for free in ever-growing databases. They constitute a remarkable resource of data distributed across more than 70 languages.

In this document, we propose B-Subtle - a novel tool for automatic creation of corpora and collection of analytical data from subtitles. Since different users might have different needs, we aim to provide a flexible system that can be fully parametrized through a configuration file. The generated corpora will serve as a knowledge base for conversational agents.

Besides the corpora generation tool, another system will be described - Say Something Deep. This system is capable creating sequence-to-sequence models to answer questions made by its users. It relies on neural networks to implement a generative approach by taking corpora generated with B-Subtle as its knowledge base.

1 Introduction

Creating a Conversational Agent (CA) for a certain domain is an accessible task considering some tools available. For instance, Pandorabots[1] is a web service for building and deploying these type of agents. However, the resulting agents have restricted knowledge due to the human effort needed to manually introduce the data.

One of the challenges is to develop methods that build sources of knowledge automatically without the need for human intervention. The foremost task is to find appropriate information sources and then extract useful data from them.

Subtitles from movies and TV shows are one of those sources. They are available on-line for free in ever-growing databases. *OpenSubtitles*[2] is one of those databases, where the number of subtitle files available surpasses the 4 million mark distributed across more than 70 languages. Considering its size and language variety, it is reasonable to say that it makes up a remarkable resource to extract valuable features (linguistically speaking) due to its broadness of covered genres and its multiple types of discourse (narrative, slang, etc.).

L2F/INESC-ID group already built a corpus from subtitles - the Subtle Corpus [Ameixa *et al.*, 2013]. It is composed by interactions - pairs of triggers[3] and answers - extracted from 6,000 English subtitle files and 4,000 Portuguese subtitle files. A tool was used to automate the process of building the corpus - we will call it Subtle tool from now on. The resulting corpus can be used as a knowledge base for a CA. However, it is limited to those two languages and it has not been updated ever since with data from new subtitles. The Subtle tool used to generated the corpus does not allow a research group to customize the corpus generation process without changing its implementation. Also, since it deals with subtitle files that have other information associated (genre of the movie, release year, timestamps of turns, etc.) it could be useful to allow collection and analysis of that data while building the corpus.

Selecting responses for a CA to give when dealing with user input consists in another challenge. The best response should be chosen or generated from the built source of knowledge. This could be accomplished by using *Deep Learning* techniques either by following a *retrieval-based* approach or a *generative* approach. *Retrieval-based* models are simpler to implement. They work with a source of predefined responses and use some kind of heuristic to pick an appropriate response. However they may be incapable of handling unseen cases. A dialogue system developed at L2F/INESC-ID group relies on a *retrieval-based* approach: Say Something Smart (SSS) [Ameixa *et al.*, 2013] uses Subtle corpus as its knowledge base. The system starts by matching the user input against a set of possible responses. Afterwards,

[1] https://www.pandorabots.com/

[2] https://www.opensubtitles.org/

[3] A turn extracted from subtitles that will cause the next one to appear - the answer.

it applies some weighted measures to give them an order. The one with the best score is returned to the user. However, when SSS needs to select the best response it is highly coupled with an internal scoring algorithm used by the search engine library[4].

On the other side, *generative* models are harder to implement but can generate entirely new answers not present in the knowledge base, although being more likely to perform grammatical errors or give irrelevant answers. Recently, an increasing number of studies have found that end-to-end CAs can be built by following purely data-driven approaches by relying on neural models. One of our main goals is to create end-to-end CAs with corpora generated by *B-Subtle* by following an approach based in recent studies.

The present document presents a revamped tool for creating corpora - *B-Subtle*. Besides the corpora generation tool, we also offer another tool - Say Something Deep (SSD). This system is capable of generating responses upon receiving some input. It relies on neural networks by using state-of-the-art Sequence-to-Sequence (seq2seq) models. Corpora generated with B-Subtle can serve as knowledge base for the conversational models created with SSD.

The remaining part of the document is organized as follows: in Section 3 we present the architectural details of our tool for building corpora - *B-Subtle*; then, in Section 4 we present our experiments where we built neural CA's by using corpora generated by *B-Subtle*; from there on we proceed to Section 5 where we describe how we evaluated our agents; finally, in Section 6 we include conclusions and future work.

2 Related Work

This section presents a review of previous work regarding existing *corpora* useful for dialogue systems and the process of generating responses with end-to-end CAs for user input in a turn-by-turn basis.

2.1 OpenSubtitles2016 Corpus

As previously pointed out, the number of subtitle files is constantly increasing everyday. The OPUS Corpus[5] have been updated in 2016 with a new dataset based on movie and TV subtitles: the OpenSubtitles2016 Corpus [Lison and Tiedemann, 2016].

This *corpus* was the ideal candidate for our research. It is the largest *corpus* of subtitles available.It has data from subtitles till the year of 2016. It also provides metadata about each subtitle file. Also it is available for multiple languages and includes 96254 files with Portuguese subtitles.

[4]Used for indexing the corpus files and retrieving results by making queries.

[5]`http://opus.lingfil.uu.se/`

2.2 End-to-end Sequence to Sequence Models

Building a CA with seq2seq involves mapping questions to responses and being able to do that with a straightforward model is very appealing. The seq2seq model can learn to map between questions and answers in either closed-domains or open-domains datasets as shown by [Lu *et al.*, 2017] and [Li *et al.*, 2015].

In [Vinyals and Le, 2015], they built a neural conversational model by using OpenSubtitles2009 [Tiedemann, 2009] [6] English subtitle files was one of the datasets used to test the model. They considered consecutive sentences as if they were uttered by distinct characters. The model was trained to predict the next sentence given the previous one. Their CA was capable of having basic fluent open-domain conversations. The model could generalize to new questions it has never seen during the training phase. However, the built CA has some drawbacks: it gives too many short and simple answers and it also lacks a way to ensure consistency during a conversation because it does not include any general world knowledge (it is an unsupervised model) and it has no memory of the past conversation. They evaluated their CA by comparing against CleverBot[7]. They asked four different humans to rate the answers given by both agents for 200 questions. Their CA achieved a better score after analyzing the results of the human evaluation. They justified the choice of using human evaluators by stating that designing a good metric to measure the quality of a conversational model remains an open research problem.

The problem with generic answers was also referred by [Guo *et al.*, 2017]. After training a seq2seq model with Cornell Movie Dialog Corpus they ended up with a high percentage of "I don't know" answers. After removing all "I don't know" sentences from the input dataset and training a new model, the responses remained vague with a high percentage of "What do you mean?"). The evaluation of the created CA was made by the team members involved in the research.

Generating long, informative, coherent and diverse responses remains a hard task. A recent review of the literature on this topic [Li *et al.*, 2015] has found that the traditional objective function that selects the best answer is unsuited for question-answering systems (although it provides state-of-the-art results in machine translation tasks). They proposed using a Maximum Mutual Information as the objective function which penalizes generic responses. They stated that more meaningful responses can be found in N-best lists given by seq2seq models but rank much lower. After applying their proposed objective function, they were able to achieve more diverse responses. OpenSubtitles2009 [Tiedemann, 2009] dataset was also used in their experiments. While training the model they used BLEU [Papineni *et al.*, 2002] for parameter tuning. They also relied on hu-

[6]One of the previous versions of the corpus presented in **??**

[7]`https://www.cleverbot.com/`

man evaluation. The judges were informed to prefer outputs that were more relevant to the preceding context, as opposed to those that were more generic. After analyzing the results, they were able to improve the number of diverse and interesting responses returned by the model. In [Shao *et al.*, 2017] similar results were obtained by using a slightly modified version of beam-search when selecting the best response. They introduced stochastic sampling operations in the beam-search algorithm. This allowed them to inject diversity earlier in the answer generation process. They also implemented a back-off strategy by choosing to fallback to the baseline model with standard beam-search algorithm when the response was shorter than 40 characters[8]. Once again, they also relied on an evaluation done by humans by asking them to rank the answers given by their model with a 5-scale rating[9]. Some of their methods were able to generate longer answers, however those had worse results after analyzing the classifications given by the judges.

3 B-Subtle – General Overview

We present a revamped tool for creating corpora - B-Subtle. We aim to replace the existing Subtle tool by providing the following features:

1. Create corpora of interactions with meta-data associated such as the genre of the movie/TV show, release year, spoken language, subtitle language, etc. This allows end-users to generate corpora that meet specific requirements: for example including only interactions collected from movies with "Action" as a genre;
2. Support OpenSubtitles2016 Corpus [Lison and Tiedemann, 2016] files as an input;
3. Support different output formats for the corpus generated such as JavaScript Object Notation (JSON) or eXtensible Markup Language (XML) files. This could be useful for end-users since they could choose the output format that fits their needs;
4. User-friendly configuration file: allow the user to fully control the behavior of B-Subtle when creating corpora simply by adjusting parameters in a configuration file.
5. Allow end-users to collect analytical data about movies, TV shows and subtitle files.
6. Deal with specific language details (e.g. encodings) in order to allow end-users to work with a broader range of languages.

We adopted an architecture of components (Figure 1) so that B-Subtle can be easily expanded in the future.

3.1 Input Files

The OpenSubtitles2016 Corpus files are in XML format, therefore a dedicated parser was implemented. To process all the data successfully, some additional steps have been performed. For example, some files contained invalid XML characters that need to be deleted so that the file could be correctly analyzed without being immediately discarded. Also, during the meta-data collection step, the "duration"[10] field of the subtitle file was found written in multiple patterns and we converted them to a unified format[11].

While analyzing multiple parser architectures for XML files we came across multiple options.We ended up choosing a Document Object Model (DOM) parsing architecture because it allowed a faster development of the B-Subtle tool.

3.2 Meta-data Collectors

These components are responsible for enriching the meta-data provided by original input files with meta-data from external sources (for instance the genre of the movie if this information is not available). They can be particularly useful for *.srt* files directly downloaded from websites, which lack on information. We have currently implemented the *themoviedb*[12] *Meta-data Collector* [13]. This component makes an Hypertext Transfer Protocol (HTTP) request to *themoviedb* with an International Movie Database (IMDb) identifier[14] and receives back a JSON that is parsed by B-Subtle. The relevant information is extracted (e.g. extracting the movie certification codes in order to classify the audience type). This information can then be filtered with the components we will describe next.

3.3 Filters

When dealing with input data that is rich in meta-data[15], filters may be applied. This feature allows end-users to easily generate targeted corpora (e.g. generate a corpus of interactions from movies with "Western" as a genre and released before the year 1990).

Two types of filters can be used: Meta-data Filters and Interaction Pair Filters, described below.

Meta-data Filters

- **Audience**: allows filtering subtitle files according to an audience rating/certification. This information can be added by our *themoviedb* Meta-Data Collector. This filter supports a flag for adult movies. It also supports filtering by motion picture content rating when provided together with a country identifier[16] (different countries have different cri-

[8]Textual length.

[9]Excellent, Good, Acceptable, Mediocre, and Bad.

[10]Refers to the total duration of the source movie related to the subtitle file.

[11]An *Integer* with the amount of time in minutes.

[12]`www.themoviedb.org`

[13]Limited to 40 API requests every 10 seconds by IP address.

[14]It is provided by OpenSubtitles2016 Corpus as a filename of the subtitle files.

[15]Possibly enriched with B-Subtle's own Meta-data Getters.

[16]Details about motion picture content rating in multiple countries: `https://en.wikipedia.org/wiki/Motion_picture_content_rating_system`

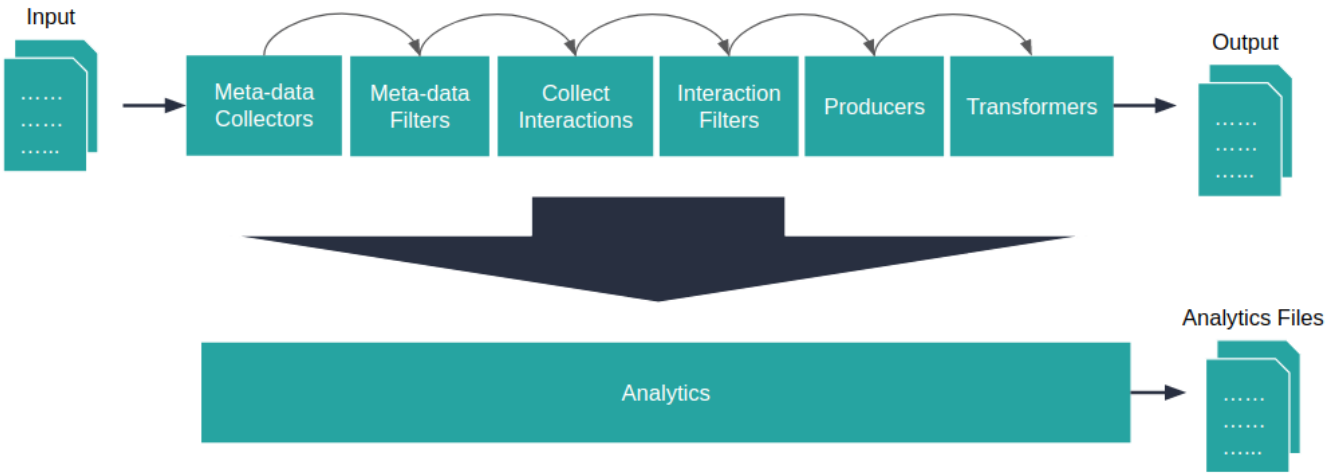

Figura 1: Possible *B-Subtle pipeline* for OpenSubtitles 2016 files using all the components available.

teria for content and age rating). (e.g. applying the flag for filtering adult movies will result in skipping those subtitle files; defining the content rating value as M/16 with Portugal as a country would result in accepting all subtitles files from movies with that content rating).

- **Country**: allows filtering subtitles of movies/TV shows made in a specific country or set of countries. Using a regular expression[17] for the country name is also supported. (e.g. accept only subtitles files from movies made in countries starting with "Po" by using the following regular expression: `"^\Po"`).
- **Country Quantity**: allows filtering subtitles of movies/TV shows made in a determined quantity of countries. A maximum, minimum or exact quantity can be defined. A range can also be used. (e.g. defining a range value of 2 to 4 would result in accepting subtitle files from movies filmed in at least 2 countries but filmed in less than 4 countries).
- **Duration**: allows filtering by the total duration of the movie (in minutes). A maximum, minimum or exact quantity can be defined. A range can also be used. (e.g. accept only subtitles from movies with less than 90 minutes by defining a maximum quantity of minutes).
- **Encoding**: allows filtering subtitle files written in a specific encoding. Using a regular expression for the encoding is also supported as well as the existence of it (one might want to filter only the subtitle files that have the encoding correctly identified).
- **Genre**: allows filtering subtitles of movies/TV shows belonging to a specific genre or set of genres. Using a regular expression for the genre type is also supported.
- **Genre Quantity**: allows filtering subtitles of movies/TV shows belonging to a determined quantity of genres. A maximum, minimum or exact quantity can be defined. A range can also be used. (e.g. accept only subtitles with "Action and "Comedy" as a genre).
- **IMDb Identifier**: allows filtering subtitle files that have the IMDb ID present in the meta-data fields.
- **Movie Title**: allows subtitle files to be filtered by movie name by providing a regular expression. The existence of that field in the meta-data can also be tested (one might want to filter subtitles that have a movie title associated, some of them might not have that information included, since this field is not provided by OpenSubtitles2016 Corpus files).
- **Original Language**: allows filtering subtitles of movies/TV shows made in a specific language or languages. Using a regular expression for the original language is also supported.
- **Original Language Quantity**: allows filtering subtitles of movies/TV shows made in a determined quantity of original languages. A maximum, minimum or exact quantity can be defined. A range can also be used.
- **Movie Rating**: allows filtering subtitle files based on the movie rating associated with the subtitles. It supports checking for the existence of that field in the meta-data as well as a maximum. minimum, exact or range of values.
- **Subtitle Rating**: allows filtering subtitle files based on the subtitle rating associated. It supports checking for the existence of that field in the meta-data as well as a maximum, minimum, exact or range of values. (e.g. accept only subtitle files with a rating above 6.3 in a scale of 0 to 10 by defining a minimum value).

[17] A sequence of characters that define a search pattern.

- **Year**: allows filtering files based on the release year of the movie. It supports checking for the existence of that field in the meta-data as well as a maximum, minimum, exact or range of values.

Interaction Pairs Filters

The following filters are available for the interaction pairs. Some of them require a *Producer* to be applied *a priori*:

- **Interaction Interval**: allows filtering the value of time interval allowed between a trigger and an answer. It supports checking a maximum, minimum, exact, or range of values (e.g. collect interaction pairs where the answer appears up to 4 seconds after the trigger);
- **Trigger/Answer Sentiment**: allows filtering interaction pairs where the trigger/answer expresses a sentiment defined by the user (requires the Sentiment Producer Component). (e.g. accepting only triggers with a positive sentiment).
- **Trigger/Answer Tokens Quantity**: allows filtering interaction pairs where the trigger/answer has a determined amount of tokens (requires the Tokenizer Producer Component). A maximum, minimum or exact quantity can be defined. A range can also be used. (e.g. accepting only answers with more than 5 tokens such as sentences like "Yes you are!" are discarded[18]).
- **Trigger/Answer Characters Quantity**: the same as the above, but for textual content length (e.g. the sentence "I am fine." contains 10 characters and if we define this filter with a minimum characters quantity of 5, that sentence is accepted if it is part of a trigger/answer).
- **Trigger/Answer Regular Expression**: allows filtering interaction pairs where the trigger/answer matches some regular expression defined by the user. This filter gives a lot of flexibility such as building a regular expression that filters out triggers containing curse words.
- **Trigger/Answer Text Content**: allows filtering interaction pairs where the trigger/answer starts with, contains or ends with some sequence of characters. The same result can be achieved by using a Trigger/Answer Regular Expression Filter, but since some users might not be advanced enough to use regular expressions we decided to provide this filter for simples use cases of text content starting with, containing or ending with some sequence of characters.

3.4 Producers

This type of components is responsible for generating additional data for the interaction pairs. The search and implementation of producers were limited to tools with existing Java libraries. We also preferred tools where Portuguese was available as a supported language.

- **OpenNLP Sentiment Analyzer**: uses a sentiment analysis tool from OpenNLP. Currently it is only prepared to deal with English sentences. It evaluates a sentences sentiment according to the following scale: very negative, negative, neutral, positive, and very positive;
- **OpenNLP Stemmer**: uses a snowball stemmer from Apache OpenNLP[19]. This stemmer supports 16 languages[20]. The language parameter is customizable through a B-Subtle configuration file (Section 3.8). By default, it is applied to both the trigger and the answer.
- **Open NLP Tokenizer**: converts the raw text from triggers and/or answers into separated tokens. It is available for Danish, German, English, Dutch, Portuguese and Sweden.
- **TreeTagger Lemmatizer**: converts the raw text from triggers and/or answers into separated lemmas. It is available for 23 languages[21].

3.5 Transformers

Transformers are entities responsible for transforming the raw text data present in the interaction pairs.

- **Lowercase**: converts the raw text from triggers and/or answers into lowercase characters.
- **Uppercase**: converts the raw text from triggers and/or answers into uppercase characters.

Some of them can also be applied to the data generated by the *Producers*.

- **Stringify Tokens**: replaces the trigger and/or answer fields by joining the tokens generated by some producer with some separator (the default is the space character).
- **Stringify Lemmas**: does the same as the above but applied to the lemmas generated by a producer.

3.6 Output Files

The generated corpora can be written to four types of output files: JSON files, XML files, Legacy files[22] or Parallel Files. Each one of those types will be described in this section. Some output types support a customizable parameter that allows the user to enable or disable pretty print (for JSON and XML), thus generating files with more or less size respectively.

[18] Contains only 4 tokens: "Yes", "you", "are" and "!"

[19] `opennlp.apache.org`

[20] Danish, Dutch, English, Finnish, French, German, Hungarian, Irish, Italian, Norwegian, Portuguese, Romanian, Russian, Spanish, Swedish, Turkish.

[21] German, English, French, Italian, Danish, Dutch, Spanish, Bulgarian, Russian, Portuguese, Galician, Greek, Chinese, Swahili, Slovak, Slovenian, Latin, Estonian, Polish, Romanian, Czech, Coptic and old French

[22] Similar to the ones generated in Subtle. Since SSS is dependent on the format of those files, this will ease the process of evaluating the system.

- **JSON Files**: we provide this output file type because JSON is just text in a standardized format. It can be useful for an end-user using our corpora files for some application that requires communication between a browser and a server. This output type supports a pretty print[23] option that can be enabled or disabled.
- **XML Files**: we provide this output file type because the way we implemented our JSON output could be adapted without much effort to output XML files. This output type also supports a pretty print option that can be enabled or disabled.
- **Legacy Files**: in order to be retro-compatible with previous systems developed at L2F/INESC-ID (e.g. SSS) we decided to include this output type. We also planned our experiments to use SSS with a corpus generated with B-Subtle. These output files are simple text files with the same fields generated by the Subtle tool.
- **Parallel Files**: this output type consists in generating at least two files: one containing the triggers and another one containing the answers. Each line of the triggers file is aligned with each line of the answers file. If we pick the same line from both files (e.g. 14th line) we get the trigger and the corresponding answer of an interaction pair. This output type allows generating corpora that can be fed to seq2seq systems (which will be the case in our experiments for generating end-to-end CAs). Two additional files can also be generated. We call them validation files (usually required for experiments with seq2seq frameworks). One is for the validation triggers and the other is for the validation answers. These files are also aligned. B-Subtle randomly samples a user-defined quantity of interaction pairs to build these validation files. The interaction pairs inserted in validation files are not present in corpus files.

3.7 Analytics

B-Subtle offers the possibility to collect analytical data about the process of creating corpora, as well as analytics about the input/output datasets. This allows to easily have information about the corpora, but also to analyze movies and TV shows information from a bunch of subtitle files (e.g. it can be interesting to study the evolution of movies pace over the years). Three types of analytical data can be generated: global, meta-data and interaction. The analytics collected are currently outputted to JSON files.

Meta-data Suppose that we want to study the pace of subtitles from 1990 till 2010 for the "Adventure" genre. We can define a Genre Filter for that type and the corresponding Year Filter with the corresponding range value and then we activate both the Global Analytics and Meta-data Analytics components. We will be able to get the average time difference between trigger and answer, giving us the information that we were interested.

Each Meta-data Filter can be configured to collect or not the analytical data. Basically, our filters will fire an event that will be captured by the Meta-data Analytics component which aggregates data received from multiple filters.

The total types of Meta-data Analytics that can be collected are as much as the total of meta-data filters available.

Interaction Pairs Aggregating Interaction Pair Analytics works similarly to the Meta-data Analytics component since we also have filters for the interaction pairs. Therefore, the total types of interaction pair analytics that can be collected are as much as the total of interaction pair filters available.

Global When generating a new corpus the end user might want to collect analytical data about the process of generating a corpus. We call it Global Analytics and it includes the following information:

- Total input files processed (includes quantity and size);
- Total invalid input files (includes quantity and size);
- Total output files generated per output type (includes quantity and size);
- Average time spent processing each file;
- Total time spent processing all files;
- Average time difference between trigger and answer;
- Average interactions pairs extracted for each input file;
- Input file with the most interactions pairs;
- Largest input file;
- Largest output file (per output type).

3.8 Configuration Files

YAML Ain't Markup Language (YAML)[24] configuration files are supported since their signal-to-noise ratio is higher without all the brackets that we are used to seeing in XML files. This makes it subjectively easier to read and edit.

For the OpenSubtitles2016 *task* the following fields are currently available to be used:

- The directory where the input files can be found;
- A list of *Meta-data Collector* components;
- A list of *Meta-data Filter* components;
- A list of *Interaction Pair Filter* components;
- A list of *Producer* components;
- A list of *Transformer* components;

[23] Show JSON with indentation in multiple lines, instead of a single line with all the information.

[24] `http://yaml.org/`

- A list of *Analytics* components;
- A list of Output components that will generated corpora files (see section 3.6).

4 Experiments

In recent experiments, the seq2seq model has been shown to be very appealing for creating CAs with purely data-driven approaches. Instead of translating from one language to another with seq2seq, we wanted to "translate" an input (trigger) to an output (answer). We developed SSD which offers a generative approach, contrasting with SSS which relies on a retrieval approach.

SSD receives a user request (the trigger) and chooses an answer. Architecturally speaking it works just like SSS: it receives an input and applies some procedures to select an output. However, the process of selecting the best answer is entirely different.

SSD relies on a seq2seq framework which is commonly used for machine translation tasks: OpenNMT-tf[25]. This framework is a general purpose sequence modeling tool built with TensorFlow[26].

We trained our conversational models by following some guidelines described in [Vinyals and Le, 2015]. We also applied to our model an attention mechanism [Luong *et al.*, 2015; Bahdanau *et al.*, 2014].

We created corpora with B-Subtle for our main experiments:

- **Corpus A:** Corpus with all Portuguese subtitles available in OpenSubtitles2016. The following B-Subtle components were used: an OpenNLP Tokenizer Producer and a Lowercase Transformer. The output resulted in Parallel files for the SSD system and Legacy Files for the SSS system. We ended up with a corpus with almost 95 million interaction pairs.
- **Corpus B:** we used the same configuration of Corpus A but added a Subtitle Rating Filter. We ended up with a corpus with almost 4,5 million interaction pairs. The validation size was set to a fixed value of 2500 interaction pairs also.
- **Corpus C:** we used the same configuration of Corpus B but added added a filter for triggers ending with a question mark. We ended up with a corpus with almost 786 thousand interaction pairs.

We also used Subtle Corpus so that we can compare our new approach to what existed before.

4.1 Creating Conversational Agents

In order to create our CAs, we configured SSD in the following way:

- **Encoder type:** Bidirectional Recurrent Neural Networks (RNN) Encoder with 2 layers, each one using Long-Short Term Memory (LSTM) as a memory cell type with 512 hidden units;
- **Decoder type:** Attentional RNN Decoder with 2 layers, each one using LSTM as a memory cell type with 512 hidden units;
- **Vocabulary Size:** 100 thousand words;
- **Word Embedding Size:** 256;
- **Training time:** for the corpus A we trained for 6 epochs due to time restrictions[27]. For the other ones we trained till the normalized loss stabilized[28].

We applied the default AdaGrad optimizer[29] provided by the OpenNMT-tf framework.

In order to try to reduce the appearance of sub-optimal answers given by our models, we did some experiments to compare a greedy search approach to a beam search approach when decoding the output. After some subjective analysis, we ended up choosing a beam search technique with a beam width of 2 and applying a length penalty of 1 (neutral). The results returned by our models with this configuration seemed to provide better overall results when compared to the others (particularly against a greedy approach).

The previous Subtle Corpus was already indexed by SSS in the past. We reused those indexes for our experiments. However, we needed to index Corpus A by following the instructions available in a *readme* file provided by the tool.

These were the agents that we created:

- **Agent Alpha**: its knowledge base comes from Corpus A and its answer selection mechanism is the SSD system with a model trained during 6 epochs;
- **Agent Bravo**: its knowledge base comes from Corpus B and its answer selection mechanism is the SSD system with a model trained during 15 epochs;
- **Agent Charlie**: its knowledge base comes from Corpus C and its answer selection mechanism is the SSD system with a model trained during 15 epochs;
- **Agent Delta**: its knowledge base comes from Corpus A and its answer selection mechanism is the SSS system;
- **Agent Echo**: its knowledge base comes from Subtle Corpus and its answer selection mechanism is the SSS system;

5 Evaluation

We aimed to compare our agents against each other so that we could examine and determine the influence of the corpora we generated as well as the architecture for generating (SSD) or selecting (SSS) the best answer. With the purpose of evaluating the performance of our agents, we asked volunteers to fill a survey. That survey consisted of a list of questions. For each question the volunteer

[25] `opennmt.net/OpenNMT-tf/`

[26] `tensorflow.org`

[27] One epoch takes almost one day and a half using two GTX 1080 Ti.

[28] When it stopped decreasing its value.

[29] Adam optimizer with clipping gradients and decay.

had to qualify an answer given by each agent in the following way:

- **Valid**: when the answer was appropriate to the subject of the question (e.g. "How old are you?" and the answer given was "I am 24 years old.");
- **Plausible**: when the answer could be appropriate in a given context (e.g. "Where do you live?" and the answer given was "We do not have time for that, keep running.") or referred to details that may belong to the person who performed the interaction (e.g. the agents could return answers containing names such as "Ok John." so that would be a plausible answer if the agent is talking with someone called John);
- **Invalid**: when the answer was not adequate or included grammatical errors that made it difficult to understand (e.g. "How old are you?" and the answer given is "Yes.").

The survey consisted in a list of 100 questions randomly picked from a set of 361 questions: 200 questions manually translated to Portuguese from [Vinyals and Le, 2015] and 161 questions made to Filipe[Ameixa *et al.*, 2014] that were already available in Portuguese. Although we refer to them as questions, some of them were commentaries (e.g. "Life is hard...").

Since some volunteers could not have the availability to fill the entire survey we decided to split it into two parts, each one with 50 questions. The first part was required and the second part was optional.

5.1 Results

A total of 44 volunteers participated in the survey. Of the participants, 32 filled both the first and the second part. Since the survey had 100 questions, each one with 5 answers (given by our agents) to be evaluated, we were able to collect 3800[30] human evaluations per agent.

We present the results of the responses given by the volunteers for the percentage of valid, plausible and invalid answers classifications given to our agents in Table 1.

The first observation is that the agents which use SSD (Alpha, Beta, and Charlie) gave answers way more preferred by our survey participants when comparing them against the agents which use SSS (Delta and Echo).

Agent Alpha achieved impressive results by having 72.97% of its answers classified as valid and 18.32% classified as plausible. Only (8.71%) of its answers were labeled as invalid. It also leads the pack by a great margin by having almost 14.44% more valid answers than the second-best Agent Charlie.

Although Charlie is the second-best agent among the five, Agent Beta showed similar behavior by having identical amounts of valid, plausible and invalid answers. Agent Charlie is only slightly better than Agent Beta, by having less amount of invalid answers (-2.63%).

[30] $(32 * 100) + (12 * 50) = 3800$

At the end of the pack, we have Agent Delta and Agent Echo. Agent Delta had the worst results by having 60.84% of its answers classified as invalid. The difference between its valid answers and plausible answers is short and the sum of the two (39.16%) does not even reach the number of valid answers given by the third-best Agent Beta (57.97%). Surprisingly, Agent Echo, which uses a smaller and outdated corpus of Portuguese subtitles (Subtle Corpus), was able to achieve better results than Agent Delta, which uses a brand new corpus created during our experiments (Corpus A).

5.2 Short and Simple Answers

While analyzing the answers given by our agents which rely on SSD (Agents Alpha, Beta, and Charlie) we noticed a high amount of short and simple responses. "Sim."(Yes.), "Não."(No.) and "Não sei."(I do not know.) were the most frequent as we can see in Table 2.

At this point, we were intrigued by the possibility of having an almost direct correlation between the percentage of short and simple answers returned by our agents and the percentage of answers marked as valid by our survey participants. As shown in Table 1 our Agent Alpha had 72.97% of its answers considered valid but 71.00% of them (Table 2) were either "Sim."(Yes.), "Não."(No.) or "Não sei."(I do not know.). We can verify similar results for Agents Beta and Charlie.

After further investigation, we identified that 23.06% of Agent Alpha valid answers correspond to responses other than the short and simple answers already described. We were able to recognize a similar pattern for the other two agents. Agent Beta got 24.86% and Agent Charlie got 28.49%. Although Agent Charlie appears to return more diverse (and perhaps more interesting) answers than its competitors for the valid label, the converse happens for the plausible label.

Despite the fact that the agents Alpha, Beta and Charlie gave a majority of short and simple answers, most of them were still considered valid by our survey participants. The short and simple answers were labeled as invalid very few times in comparison with all the other invalid answers. The high amount of short and simple answers can also be justified by the content of the questions that were chosen randomly for the research. Some of them really require a "Sim."(Yes.), "Não."(No.) or "Não sei."(I do not know.) as an answer (e.g. "Gostas do teu trabalho?"(Do you like you work?) can be satisfied by a "Não."(No.)).

The "Não sei."(I do not know.) answer seems to be the biggest issue that we identify in our SSD agents. That type of answer can make a *chatbot* boring in a conversational context. Indeed, the answer "Não sei."(I do not know.) was considered valid or plausible a significant amount of times by the volunteers who participated in our survey.

Table 1: Percentage of valid, plausible and invalid answers classifications given to our 5 agents (Alpha, Beta, Charlie, Delta, and Echo) by the volunteers who have filled out the survey.

	Agent Alpha	**Agent Beta**	**Agent Charlie**	**Agent Delta**	**Agent Echo**
Valid Answer	72.97%	57.97%	58.53%	22.58%	26.08%
Plausible Answer	18.32%	19.39%	21.47%	16.58%	20.95%
Invalid Answer	8.71%	22.63%	20.00%	60.84%	52.97%

Table 2: Amount of "Sim."(Yes.), "Não."(No.) and "Não sei."(I do not know.) answers returned by Agents Alpha, Beta and Charlie for the 100 questions included in the conducted survey.

	Agent Alpha	**Agent Beta**	**Agent Charlie**
"Sim."(Yes.)	19.00%	21.00%	18.00%
"Não."(No.)	17.00%	10.00%	10.00%
"Não sei."(I do not know.)	35.00%	24.00%	29.00%
Total	71.00%	55.00%	57.00%

6 Conclusions

6.1 Contributions

In this paper we presented both a hands-on project and a scientific research assignment. By analyzing the limitations of Subtle Corpus and Subtle Tool created by L2F/INESC-ID group, we aimed to build a brand new tool for generating *corpora* from movies and TV shows subtitles. We called it B-Subtle. Furthermore, we scrutinized the system behind *Filipe* (a CA also built by L2F/INESC-ID group), which relies on SSS for retrieving answers from Subtle Corpus after receiving a user input. After finding its limitations, we came across two possibilities: improve the existing system or develop a competing system by following a different approach based on answer generation. We opted for the latter by creating SSD, a system that uses seq2seq learning with neural networks for building models with the capability to generate answers given a user input.We believe that B-Subtle allows research groups to do more with the data available in movies and TV shows subtitles. Not only it allows them to generate corpora that meets specific requirements (e.g. through filtering or by applying modifiers that will attach more data) but it also provides them with a tool that can extract other useful data such as analytics (e.g. see the evolution of the amount of sentences by movie genre). Since subtitles are available in a wide range of languages, it enables the possibility of creating knowledge bases for CAs with different nationalities.

We created our own conversational models with a system we denominated as SSD. In our experiments we ended up creating 5 CAs. Three of them were created with SSD, thus being able to generate answers. The other 2 were created with SSS, thus using a retrieval strategy for selecting answers. We relied on our B-Subtle tool to create the knowledge bases of our agents, by creating multiple variations of Portuguese *corpora*. We were able to compare all CAs with human evaluation. The results obtained from that evaluation showed that generative agents created with SSD gave answers way more preferred than retrieval agents created with SSS.

6.2 Future Work

We are confident that our research will serve as a base for future studies on creating CAs using seq2seq neural networks. Although the results of our survey reveal that agents created with SSD have a much higher percentage of valid answers than the ones created with SSS, we found ourselves observing a problem already known in the literature: the generation of safe, short and simple responses. In fact, those type of responses seem to be appropriate most of the times (e.g. for "yes" and "no" answers), but after analyzing the answers given by SSD agents we identified almost one-third of its answers being "Não sei."(I do not know.). The human evaluation results have shown that the survey participants had a slightly more divided opinion when classifying that type of answer as valid, plausible and invalid. The problem relies on the simplicity of the seq2seq model because its objective function being optimized does not capture the actual objective of a conversation with a human. Recent studies have proposed alternatives to this objective function. In [Li *et al.*, 2015] they suggest using an objective function that avoids favoring responses that have high probability by making a tradeoff between the input given and the possible responses and vice-versa. However, they applied their new objective function *a posteriori* to an N-best list returned by the seq2seq. A similar experiment can be made by returning N-best answers of our models generated with SSD and then applying the similarity metrics that are part of SSS. In [Shao *et al.*, 2017] they changed the behavior of the beam search algorithm and introduced stochastic sampling operations. As we can see, there are multiple approaches that can be taken to reduce the number of commonplace answers.

We are also aware that our agents built with SSD lack a way to ensure consistency in a conversational context since they rely on purely unsupervised models. In our survey, we asked the participants to evaluate each pair of question-answer independently, so further evaluation should be made in a fully conversational context.

Training our models with SSD consisted in assigning values to a considerable amount of parameters related

to the configuration of the neural networks. The value of each one of them can affect in different ways the results obtained. We followed the same configuration found in similar experiments and then tweaked some of them to adjust to our reality. Still, we are not sure which configuration combination might yield better results for the *corpora* we used as input. In order to know that, we would need another neural network to learn how to train the SSD neural networks. It is a very challenging problem that can be studied with Automatic Machine Learning (AutoML) techniques.

Regarding B-Subtle, it was built with expandability in mind so we expect additional components to be added by future developers as well as support for other types of input data, for instance, Cornell Movie-Dialogs. Currently, the most evident limitation of B-Subtle is the use of The Movie DB Meta-Data Collector, because it makes HTTP requests to get data and a rate limiting policy is applied (40 requests every 10 seconds by Internet Protocol (IP) address). However, at the time of writing, we are unaware of a better alternative. It is important to note that in our preliminary and main experiments we did not use this Meta-data Collector since the currently supported input data (OpenSubtitles2016 Corpus files) already have meta-data included.

References

[Ameixa *et al.*, 2013] David Ameixa, Luísa Coheur, and Rua Alves Redol. From subtitles to human interactions: introducing the subtle corpus. Technical report, Tech. rep., INESC-ID (November 2014), 2013.

[Ameixa *et al.*, 2014] David Ameixa, Luisa Coheur, Pedro Fialho, and Paulo Quaresma. Luke, i am your father: dealing with out-of-domain requests by using movies subtitles. In *International Conference on Intelligent Virtual Agents*, pages 13–21. Springer, 2014.

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

[Guo *et al.*, 2017] Pinglei Guo, Yusi Xiang, Yunzheng Zhang, and Weiting Zhan. Snowbot: An empirical study of building chatbot using seq2seq model with different machine learning framework. 2017.

[Li *et al.*, 2015] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055*, 2015.

[Lison and Tiedemann, 2016] Pierre Lison and Jörg Tiedemann. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles. In *Proceedings of the 10th International Conference on Language Resources and Evaluation*, 2016.

[Lu *et al.*, 2017] Yichao Lu, Phillip Keung, Shaonan Zhang, Jason Sun, and Vikas Bhardwaj. A practical approach to dialogue response generation in closed domains. *arXiv preprint arXiv:1703.09439*, 2017.

[Luong *et al.*, 2015] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[Papineni *et al.*, 2002] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[Shao *et al.*, 2017] Yuanlong Shao, Stephan Gouws, Denny Britz, Anna Goldie, Brian Strope, and Ray Kurzweil. Generating high-quality and informative conversation responses with sequence-to-sequence models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2210–2219, 2017.

[Tiedemann, 2009] Jörg Tiedemann. News from opus - a collection of multilingual parallel corpora with tools and interfaces. In *Recent advances in natural language processing*, volume 5, pages 237–248, 2009.

[Vinyals and Le, 2015] Oriol Vinyals and Quoc Le. A neural conversational model. *arXiv preprint arXiv:1506.05869*, 2015.