What's The Answer: Dialogue Annotation

809 | Supervised Project | Report

María Andrea Cruz Blandón Gosse Minnema Aria Nourbakhsh

May 28, 2018

Co-funded by the
Erasmus+ Programme
of the European Union

Contents

1. Introduction

1.1. Overview and motivation

This report presents the results of our 'Supervised Project', carried out as part of the first year of the Cognitive Science master (NLP track) at the Université de Lorraine, which was also our first year in the Erasmus Mundus Joint Master's Degree in Language and Communication Technologies. The project, supervised by Maxime Amblard and Maria Boritchev, focused on identifying and classifying question-answer pairs in dialogues. We had three main objectives:

1. Design a classification schema for questions and answers;
2. Write an annotation guide and use it to manually annotate several dialogues;
3. Explore machine learning approaches to automate annotation.

Dialogues are at the heart of human communication, and it is hard to imagine a dialogue without questions or answers. At the same time, questions, and their corresponding answers, are a very complex topic, involving all levels of linguistic analysis, from phonology and syntax to semantics and pragmatics. Questions and answers are also of considerable interest for the field of NLP (Natural Language Processing), for example in tasks such as building automated question answering systems and discourse modeling. Hence, a better understanding of questions and answers would not only help developing better theories of human language, but also has many potential technological applications.

This project aims to contribute to this in two main ways: first of all, by designing our annotation schema in a corpus-based way, we hope to get a more realistic sense of what types of questions and answers are 'out there' and how they can be identified. Moreover, by writing an annotation guide, and exploring methods to automate question and answer tagging, we provide new practical tools for both language researchers and NLP engineers for producing annotated corpora.

1.2. Research context

Our project was carried out in the context of two separate, but closely related lines of inquiry that are currently being pursued by the Sémagramme group at Loria:[1] dynamic discourse modeling, and the project *Schizophrénie et Langage: Analyse et Modélisation* (SLAM, 'Schizophrenia and Language: Analysis and Modeling'). Dynamic discourse modeling analyzes dialogues in terms of representations of the information in the discourse that are updated as new utterances are added. Theories of this kind were first developed in the 1980s (see the introduction section of [4] for a brief historical overview); a modern approach developed at Loria is Type-Theoretic Dynamic Logic (TTDL) [9]. Meanwhile, the SLAM project investigates the language and discourse of patients with schizophrenia [1, 24]. Conversations of people with schizophrenia are (or appear to be) characterized by many semantic and pragmatic inconsistencies. [24] and later

[1] Loria (*Laboratoire Lorrain de Recherche en Informatique et ses Applications*) is a research institute located in Nancy, France. See www.loria.fr and http://www.loria.fr/en/research/teams/semagramme/.

work (see [32, 4]) have tried to analyze such inconsistencies in terms of dynamic frameworks such as TTDL.

Our work aims to contribute to these projects in two ways. First, our annotation system for questions and answers could be used as a preprocessing step for automatic dialogue modeling systems: our annotations can tell such a system where the questions and answers in the dialogue are, and also give it a global idea of what kind of information these questions and answers contribute to the discourse. The dialogue modeling system can then further process this information to identify the precise semantic and pragmatic contribution and relate it to previous information. Second, by providing a way to analyze questions and answers in unimpaired ('normal') speech, we create a baseline for further investigations into how the questions and answers of patients with schizophrenia (or other forms of impaired or 'abnormal' language) might differ from this.

1.3. Report structure

The remainder of this report is divided into three main parts, each one corresponding to one of the objectives listed in section 1.1, followed by a general conclusion. The first part, chapter 2, begins by describing and motivating our annotation schema. Next, in chapter 3, we describe how we experimented with applying this schema to data from various corpora and with writing an annotation guide for describing this process as explicitly as possible. In section 3.1, we describe the experiments themselves, and in section 3.2, we analyze and evaluate our results. We also describe how we have used this evaluation to improve our annotation guide and establish a definitive version of our 'golden standard' corpus, for use in our machine learning experiments.

The third main part of our report (chapter 4) describes our experiments with trying to automate the annotation process using different types of machine learning techniques. First, in section 4.1, we define and motivate the annotation task as a classification problem, before introducing experiments using two different types of approaches—traditional statistical techniques, and more modern neural network approaches—for solving this problem. After this, in section 4.2, we present and evaluate the results of these experiments.

Finally, in chapter 5, we look back on our project as a whole, summarize our findings, and sketch how our work could serve as a basis for future research.

2. Annotation schema

In this section, we present our schema for classifying question-and-answer pairs. First, in section 2.1, we provide a global overview of the schema: we explain the general philosophy behind it. Then, in section 2.2, we briefly describe and define the schema and the tags used in it. As previously mentioned, one purpose of this study is to write an annotation guide. The annotation guide contains examples and usage of our schema; the reader is referred to it for more information (appendix B).

2.1. Annotation Overview

The main objective of our schema is to provide a way of classifying question-and-answer pairs that is useful for discourse modeling. Furthermore, it should find a balance between doing justice to the complexity of the data and the simplicity to be used by both humans and machines in a consistent way. This has a number of consequences for both the linguistic nature and the level of detail of the classification.

First of all, in order to be useful for discourse modeling, the schema should take into account not just the form of questions and answers, but also their semantics and pragmatics. At the same time, given that semantic and pragmatic features are harder to rigorously define, cues from syntax (and to a lesser extent, prosody), need to be used wherever possible. This means that each of the categories used in our schema is necessarily a combination of syntactic, pragmatic, and semantic features. Moreover, for the purpose of discourse modeling, the schema should ideally be as fine-grained as possible and indicate not only the exact location and span of a question or answer, but also precisely define how it contributes to the discourse. However, a schema with a very high level of detail would imply a considerable time investment for human annotators and requires a more complex annotation structure, which increases the likelihood of mistakes and also makes it harder to apply machine learning approaches. Our schema tries to define the span and discourse contribution of questions and answers as precise as possible while limiting itself to criteria that both human annotators and machine learning algorithms should, at least in principle, be able to consistently apply.

For designing the schema, we used a corpus-driven approach wherever possible. In more practical terms, this means that we started with minimal assumptions about the questions and answers present in the corpora that we have annotated (see section 3.1.3 for information about these corpora), and then simply began annotating the corpora, revising and expanding our schema as many times as necessary along the way. Also, most of the examples presented in the following sections belongs to the corpus we used for our study.

Questions

For designing our classification scheme for questions, we started with the assumption that the corpora would contain at least two well-known and well-defined categories of questions: yes/no questions and wh-questions [13]. In our opinion, both of these types are useful *a priori*, because they are each associated with a clear set of syntactic, semantic and pragmatic characteristics. Prototypically, in English, a yes/no question is characterized by subject-auxiliary

inversion and do-support (syntax), expresses a proposition that could be true or false (semantics), and its answer is expected to either confirm or deny this proposition (pragmatics). On the other hand, a prototypical English wh-question contains a fronted constituent that starts with a wh-word (syntax), expresses a proposition with missing information (semantics), and expects the answerer to supply this missing information (pragmatics) [13].

Starting from this assumption, we looked for questions in our corpora that did not correspond to either of the two prototypes and tried to extend our schema accordingly. We found that the questions that did not have the prototypical characteristics of either a yes/no-question or a wh-question diverged from those prototypes in one of several ways. First of all, there are questions whose syntactic form does not correspond to that of either prototype. This group of questions includes shortened questions that consist only of a single word (1) or a single syntactic phrase (2), and also yes/no questions that lack subject-auxiliary inversion (3) or wh-questions whose wh-constituent is not fronted ('wh-in-situ' questions) (4). Since the semantics and pragmatics of this type of 'deviant' questions is very similar to those of prototypical yes/no and wh-questions, we decided to add no new categories for it. Although this decision helps to keep the schema simple, it has the disadvantage of making the two original categories, yes/no questions and wh-questions, somewhat less straightforward to define.

(1) Go? (SCoSE/Amy, 208)

(2) What for? (SCoSE/Amy, 495)

(3) So you're gonna be home then? (SCoSE/Amy, 33)

(4) Some other stuff like what? (SCoSE/Amy, 2914)

A second group of questions has the syntactic characteristics of a yes/no question or a wh-question, but whose pragmatics and/or semantics is different. For example, in a question like (13), the asker of the question suggests a way to complete the utterance of the previous speaker, and the expected answer would confirm or deny this suggestion. This is subtly different from a prototypical yes/no question because the asker of the question does not so much ask their interlocutor to confirm the truth value of the suggestion, but rather if the suggestion corresponds to what the previous speaker intended to say. For the purpose of discourse modeling, it might be useful to classify this kind of question as a separate type.

(5) A: And Amanda ...
B: Making an evening of it? (SCoSE/Amy, 23-24)

Other questions with a divergent pragmatics prompted a more fundamental revision of the classification scheme:

(6) Yeah, do you really? (SCoSE/Amy, 217)

(7) Who wants to work? (SCoSE/Amy, 3427)

These questions appear to be a yes/no question or a wh-question, respectively, but their context and intonation makes clear that the person who asks them is not actually interested in the confirmation or denial of the proposition (6), or in the set of people who would like to work (7). Instead, such questions can have various so-called *phatic functions*. The term 'phatic' was coined in the early 20th century by the anthropologist Bronisław Malinowski, who used it to refer to speech that is not 'used primarily to convey meaning, the meaning which is symbolically theirs' but that 'fulfil[ls] a social function' and is an 'act serving the direct aim of binding hearer to speaker by a tie of some social sentiment or other' [16, pp. 314-316] (cited in [28]). In the context of question classification, [13] defines 'phatic information' questions as questions that enable 'the speaker to check that the hearer is following the information

exchange in the conversation and/or is aware of the relevant background information'. In our classification schema, phatic questions have a slightly broader meaning; the category includes any question that is used by the speaker to indicate that they are still paying attention, to express surprise, or in a rhetorical way.

Finally, there are questions that are semantically and pragmatically similar to wh-questions, but are syntactically closer to yes/no questions:

(8) Is it a guy or a girl? (SCoSE/Amy, 2296)

This kind of question, like yes/no questions exhibits subject-auxiliary inversion, but does not ask for the confirmation or denial of the proposition that it expresses.[1] Instead, it expects the answerer to provide some missing information, just like a wh-question (the equivalent would be something like 'what gender has this person?'), with the only difference that the set of options to choose from is made explicit in the question.

The existence of questions with either a form or a function (but not both) that corresponds to that of a yes/no question or of a wh-question led us to split up our question schema into two dimensions: 'form' and 'function'. In this way, we can do justice to the facts that questions like (13) or (6) have the same form as a yes/no question but have a different, specific function, and that (8) is functionally similar to a wh-question even though its form is different.

2.1.1. Features

In order to annotate the semantic contribution of wh-questions and disjunctive questions (and their corresponding answers) in a more precise and more useful way, our annotation schema does not only categorize the type of the question itself, but also that of the missing information that the question asks about. More formally speaking, the 'missing information' in a question has a particular semantic role. In the framework developed in Maria Boritchev's thesis [4], these semantic roles (called 'features') are an important part of the representation of the situation ('frame') that is denoted by the discourse. In order to be compatible with her work, we use the same terminology and also the same feature set (see the 'Features' paragraph in section 2.2) as in Boritchev's framework.

2.1.2. Complexity and reported statement

There are two other phenomena that we believe are important to include in our annotation schema. The first of these concerns the complexity of questions: in some cases, a single utterance asks two questions at the same time; in such cases it is important to mark in our annotations that these questions are related. Secondly, discourse sometimes includes quoted speech (the discourse participants are repeating someone else's words); for the purpose of discourse modeling, questions that are part of such quoted speech have a different status than 'normal' questions, since they are effectively part of a different discourse. Hence, it is import to mark whether or not a question is quoted this also apply to quoted answers.

2.1.3. Answers

For constructing a classification schema for answers, we used the question types that we defined as a basis, and then examined what groups of answers were associated with each question type. We started by finding prototypical answer types and then revised the schema based on

[1] Of course, it is possible to interpret questions containing a disjunction in this way ('is it the case that this person is either a boy or a girl?'), but in this case this is obviously not the intended interpretation.

deviant answers that were found in the corpora. For example, an answer to a yes/no question prototypically either confirms or denies the proposition contained in the question, and an answer to a wh-question prototypically provides the missing information expressed by the wh-constituent in the question.

Again, non-prototypical cases can be divided into several categories. First of all, there are answers expressing uncertainty:

(9) Uhm, I don't even know for sure. (SCoSE/Amy, 1265)

A more interesting type of answer denies the presuppositions (assumptions) that are implicit in the question:

(10) A: What are your plans now?
B: I don't have any plans. (SCoSE/Amy, 241-242)

A different type of answer expresses information that, at least at first sight, is not at all related to the information that the question asked for:

(11) A: When will you guys get off?
B: My last exam is like ... I don't know. (SCoSE/Amy, 243-244)

Finally, there are answers with little to no semantic content and have a phatic function:

(12) A: You know?
B: Oh yeah. (SCoSE/Amy, 900-901)

Our final schema defines a separate answer type for each of these categories of deviant answers.

2.2. Tag descriptions

As a result of previous explanation of different form and function and the way we defined our categories to annotate, we needed a set of tags for annotation of each of the previously explained phenomena. A major part of the annotation guide was defined the precise set of tags. In this report we will provide a brief definition of those tags. Please see the annotation guide (appendix B) for more information.

2.2.1. Question types

In this section we list all the tags that we used to annotate the different types of questions, according to the classification based on form and function of the question, see section 2.1.

Table 2.1 summarizes the type of questions according to their form and function and the tags we assigned for each type.

An important point is that sometimes distinction between different types of questions are not clear cut and an annotator would have difficulties in deciding the assignment of the appropriate tag for a question.

2.2.2. Features

As specified, both the wh-question and disjunction question express a proposition with missing information which has a particular semantic role and the expected answer should contain a constituent with the same role. For example, in '*When* did you leave the party?', the wh-phrase 'when' denotes the time when the addressee left the party, and an appropriate answer

Tag	Name	Form	Function
YN	Yes/no question	Inversion, do-support	Asks for the truth value of a proposition
WH	Wh-question	Wh-constituent, fronting	Asks for a feature (see feature table)
DQ	Disjunctive question	Contains a disjunction (”or”)	Asks for a feature (see feature table)
PQ	Phatic question	Any form	Phatic function
CS	Completion suggestion	Any form	Complete the previous speaker’s utterance

Table 2.1.: Question tags

to this question (e.g. ‘around midnight’) should also have this semantics. As we said earlier we adopted the list of features used by [4] in our work. The list of tags for each feature is as follows.

Tag	Name
TMP	Temporality
LOC	Location
AG	Agent
TH	Theme
OW	Owner
RE	Reason
CH	Characteristic

Table 2.2.: Features

The following table shows the corresponding semantic role for each wh-word. Among these words, *what* and *which* are followed by a phrase which can ask for any one of the features.

Wh- word	Features
What + focus phrase	Feature
When	Temporality
Where	Location
Who	Agent
Whom	Theme
Which + focus phrase	Feature
Whose	Owner
Why	Reason
How	Characteristic

Table 2.3.: Feature pairs

2.2.3. Complexity and reported statement

In our work, there are other information about question and answers that worth to be annotated. The first one is when a question or answer is a reported statement. We call these type of questions as 'quoted questions', although the tags contain the word *question* the phenomena was also presented in answers and since these tags were already used we continued using them for answers. Another interesting phenomena is when a statement contains two or more questions. To differentiate between these type of statements with simple questions, we introduced two other tags. The list of these tags are in table 2.4

Tags	Name
NQ	Non-quoted Question
QQ	Quoted Question
SQ	Single Question
MQ	Multiple Question

Table 2.4.: Other tags

2.2.4. Answers

In a conversation, usually, a question is followed by an answer. It can be intuitively observed that some type of questions restrict the type of answers they require. For defining every type of answer, as our corpus based approach dictates, we looked for question types and its corresponding answer. Accordingly, we specified the following tags for the answer types.

In the table 2.5 type of answer it should be noted that UT is different from not assigning tag to an answer. An unrelated topic is an indirect way of answering a question. There may be some logical or discourse related relation between the question and answer, and this differ from a statement that is completely unrelated to a question. Among these types of answer, there may be overlaps. For example, a deny the assumption answer can be thought of as a negative answer because it is possible that they share the same grammatical and semantic structure. Different factors including the context and prosody are relevant to decide overlapping tags.

Tags	**Name**	**Question Type**
PA	Positive Answer	YN, CS
NA	Negative Answer	YN, CS
FA	Feature Answer	DA, WH
PHA	Phatic Answer	YN, CS, DQ, WH, PQ
UA	Uncertainty Answers	YN, CS, DQ, WH, PQ
UT	Unrelated Topic	YN, CS, DQ, WH, PQ
DA	Deny the Assumption	YN, CS, DQ, WH, PQ

Table 2.5.: Answers

3. Annotations

3.1. Annotation experiments

Having defined the annotation schema described in the previous section, we have experimented with applying the schema on real-world data. Our experiment consists of two parts: annotating (parts of) several conversational corpora in different languages, and writing an annotation guide to explicitly define the annotation process for ourselves and future annotators. In this section, we give an overview of these experiments: first, in 3.1.1 we explain what software we used; in section 3.1.2 we describe our annotation guide; finally, in section 3.1.3, we explain which corpora we used for our experiments.

3.1.1. Software

The software that we used for annotating our corpora is ELAN [18]. The main advantage of ELAN is that it can read and convert the original formats of the corpora (CLAN in the case of the SCoSE corpora, and Praat TextGrids in the case of the CGN corpus). Another advantage is that it allows the annotator to create several layers (or 'tiers') of annotations. In this way, each fragment of the corpus that we would like to annotate can be aligned with a question type tag, a feature tag, etc. This multi-layer approach also enables assigning a single annotation to multiple fragments of the corpus (for example if a question is spread over several utterances) or tagging only part of an utterance.

An important characteristic of ELAN is that it is primarily intended to annotate video and/or audio files. For our purposes, this is both an advantage and a disadvantage. On the one hand, in ELAN it is possible to view and edit the transcriptions and the original audio of the corpora at the same time. This makes it easier to check, for example, the prosody of a given sentence, which is sometimes vital for assigning the appropriate question type tag to it. On the other hand, however, ELAN's focus on media files makes it harder to export our final product and prepare it for automatic evaluation and machine learning. For our final product, we are primarily interested in the text of each questions or answer, together with the annotations we assigned to them (type, complexity, etc.); references to the original media files are less relevant at this point. However, ELAN's XML-representation of the annotations is tightly linked to these media files, and defines transcriptions and annotations only in a temporal way (i.e. for each transcription or annotation it records its starting and ending time in the original media file), not in a logical way (i.e. in terms of relationships between annotations on different layers). This means that they are harder to read and require some post-processing to extract the text of the questions and answers that were annotated and connect these text fragments to the corresponding annotations.[1]

In order to do this post-processing (which, at the same time, is also a preprocessing step for the agreement evaluation and machine learning algorithms), we have developed a script that reads the ELAN files of the annotated corpora, finds the time spans of all text fragments of the original transcriptions as well as of all annotations, tries to link these together, and outputs the results in a structured (JSON) format that connects transcriptions and annotations in a logical,

[1] This problem might have been partially avoided by ordering tiers hierarchically; however, doing so would also have made the annotation work more complex and time-consuming than it was under our current approach.

rather than a temporal way. Although this is mostly straightforward, one problem is that in many cases, there is a temporal overlap between a question and its answer. If this overlap is too large, it is difficult for the post-processing algorithm to decide to which utterance a given annotation should be connected. In this case, the script adds a special flag in the output file which signals that manual correction might be needed.

3.1.2. Annotation guide

Objectives

This project is the beginning of an initiative to annotate natural language dialogues. One of our aims is to provide the future works with a concise, clear and precise guideline for annotation question and answers. For this purpose, we wrote an annotation guide (see appendix B) to guide an annotator with details, examples and software use, so that someone who has little theoretical background in this particular field can use it and produce annotations.

Tag definitions

In the annotation guide, after explaining the annotation software and how to obtain appropriate corpora for this task, we explained question types. For each question type some prototypical examples were provided and the tags that should be used were defined as well. At the end of our annotation guide, we included a part for troubleshooting. The aim of this part is to address some of the possible problems that an annotator may encounter in the process of annotation.

Annotation procedure

In the annotation guide we included a part for annotation procedure. The annotation procedure guides the annotator to find the question and answers, deal with corpus mistakes in transcription, determining the question type, determining complexity and indirectness and assigning feature types. Each part of the annotation procedure is provided by a heuristic algorithmic way of determining the appropriate tags for every information that we defined for this task.

As it is described in 2.1 there are shared forms between different type of questions. That makes the automatic and semi-automatic classification a difficult task. If we consider at once all the possible types of questions that share the same form, it is relevant to define a precedence order, to reduce misclassification. For this purpose, we defined a precedence order from the most general to the most specific type of question, that is from questions with easily identifiable characteristics, such as wh-questions which contain a wh-phrase to those that can have different forms as it is the case of phatic questions. The following list is the precedence order we used in our schema:

1. Wh-questions
2. Disjunctive question
3. Yes/No questions
4. Completion suggestion
5. Phatic questions

3.1.3. Annotated corpora

The corpora that we annotated can be divided into two sections: an English-language 'golden standard' corpus, and three 'experimental' corpora (in Spanish, Dutch, and English). Annotation of the golden standard corpus was a joint effort; each of the three members of our group annotated around a quarter of the corpus individually, and the remaining part was annotated by all three members and used for analyzing inter-annotator agreement (see section 3.2.2). The experimental corpora were divided between the three members; the purpose of annotating these corpora was to test our classification schema on languages other than English (for the Dutch and Spanish corpora) and also to produce more English data for use in our machine-learning experiments (see chapter 4).

Selection criteria

Our work focuses on annotating questions and answers in natural, spoken conversation. The SLAM project [1] and related projects, in the context of which we completed our study, are primarily interested in discourse that is in some way 'not normal', such that of patients with schizophrenia and that of young children. However, for the present study, we decided to focus on 'normal' discourse, so that we can get an impression of how well our schema works before applying it to more difficult datasets.

Hence, when selecting corpora for use in our study, we used the following criteria:

- The data should be natural discourse (i.e. spontaneous, rather than elicited, conversations);
- The participants in the discourse should be unimpaired, adult speakers;
- There should be transcriptions available in a uniform format (so that the same annotation tools could be used for all of the data).

Corpora description

Based on these criteria, we decided to use the following corpora:

- **SCoSE (Saarbrücken Corpus of Spoken English)** The SCoSE project [11] is a corpus of spoken English compiled at Saarland University. It is part of the CABank section of TalkBank project. We only used Part 1 of the corpus, which is part of the TalkBank project. TalkBank is a collection of corpora in different languages that can be used for various kinds of communication research. SCoSE (Part 1) contains several corpora for use in conversational analysis (CA).

  SCoSE Part 1 is approximately 45000 words[2] and consists of several transcriptions of conversations between college students in Illinois (US) that took place in an informal setting. We annotated the following conversations in this part of the corpus:

  1. The conversation *"Amy"* (~25000 words) was used as the 'golden standard' corpus;
  2. The conversation *"Mary"* (~11000 words) was used as part of our 'experimental' corpus, and as a test dataset for our machine learning experiments.

- **CallFriend** The CallFriend corpus [5] is, like SCoSE, part of the CABank. The corpus was originally compiled for use in the development of language identification technologies. The project was primarily led by the Linguistic Data Consortium. Each of the participants

[2] All word counts listed here were obtained using the Unix command line tool `wc`.

in the corpus got a free call of 30 minutes and were allowed to talk to anyone they wanted, usually a relative or a close friend, and speak about any topic.

The conversation transcriptions that we annotated (as part of our 'experimental' corpus) are *"2034h"*, *"2128h"*, and *"4053h"* (around 20000 words in total). The participants in these conversation are people from Colombia.

- **Corpus Gesproken Nederlands (CGN)** The CGN [20] is a joint project of the Dutch and Flemish governments. It was first published in 2004 and consists of 800 hours of recordings of spoken Dutch. The corpus contains various kinds of spoken language; the conversation transcriptions that we used for our annotations come from the section of the CGN that contains spontaneous phone conversations.

  As part of our 'experimental' corpus, we annotated three conversations from this corpus, all of them between people in their 20s and 30s: *"fn008000"*, *"fn008001"*, and *"fn008003"* (around 7000 words in total).

3.2. Annotation results and evaluation

After annotating the 'golden' and 'experimental' corpora described in the previous section, we analyzed and evaluated our own work. In this section we present this analysis: first, in 3.2.1 we present some descriptive statistics that give a general overview of our annotation work. Then, in 3.2.2, we analyze inter-annotator agreement of our annotations on our 'golden' corpus; in 3.2.3, we summarize our qualitative analysis of the disagreements that were found, and explain how we used this analysis to establish a definitive version the 'golden standard' and to improve the annotation guide. Finally, in 3.2.4 we briefly discuss the language-specific issues that arose during annotation of the two non-English corpora.

3.2.1. Descriptive statistics

A general overview of our annotation work is given in Table 3.1. It shows at least two interesting patterns. First of all, we see that the question/non-question ratio is around the same in each of the three corpora: about 5-10% of the lines that we annotated was tagged as a question. Furthermore, we also see that, in every corpus, there are more questions than answers (which is a logical consequence of the fact that every answer needs to be preceded by a question, but that not every question is actually answered), and that the number of questions that is answered varies between a little over 60% (Dutch) and more than 80% (Spanish).

Tables 3.2-3.6 and figures 3.1-3.6 show the distribution of each of the tags in our annotation schema, grouped by language. There are some differences between the languages, but in most cases, the global pictures are quite similar. For example, for question types (table 3.2, figure 3.1), the most common tag in each language is YN (yes/no question), followed by WH (wh-question) or PQ (phatic question). However, exactly how prevalent YN is depends on the language: in the Dutch corpus, it accounts for almost 60% of the questions, while in Spanish and English corpora this is about 40%.

In the distribution of answer types (table 3.3, figure 3.2), there are several obvious patterns: for example, the Spanish and Dutch corpora have a much larger percentage of FA (feature answer) tags than the English one; this difference is caused by the larger number of WH-questions in these two languages. Something that is shared between all the three corpora is that in call cases, there are far more positive answers than negative answers. Interestingly, however, this difference is much larger in the Dutch corpus than in the other languages.

For feature types (table 3.4, figure 3.3), an interesting pattern is that in all the corpora, TH (theme) is by far the most common tag. One possible contribution is that in the cases that the

Corpus	Language	No. Lines	No. Questions	No. Answers
Amy	English	3580	238	183
Mary	English	1383	184	106
Total (English)	*English*	*4963*	*422*	*289*
fn008000	Dutch	204	28	24
fn008001	Dutch	338	33	29
fn008003	Dutch	393	26	19
Total (Dutch)	*Dutch*	*935*	*87*	*72*
2034h	Spanish	921	51	35
2128h	Spanish	937	73	57
4053h	Spanish	790	68	30
Total (Spanish)	*Spanish*	*2648*	*192*	*122*

Table 3.1.: Summary of the annotation experiments

feature tag is not clear, we specified in the annotation guide to use TH as the default value. But for the other features that have zero presence in the corpora, one should not generalize it to any language as the the corpora that were tagged are relatively small. Finally, when it comes to the distribution of complex answers and of quoted questions and answers, the situation is almost identical for all corpora. Both quoted questions and answers, as well as complex questions, are very rare.

	CS	DQ	PQ	WH	YN
Spanish	0.010638	0.015957	0.244681	0.329787	**0.398936**
Dutch	0	0.011494	0.080460	0.264368	**0.643678**
English	0.016588	0.011848	0.315166	0.234597	**0.421801**

Table 3.2.: Distribution of question types

	DA	FA	NA	PA	PHA	UA	UT
Spanish	0.017699	0.309735	0.132743	**0.371681**	0.061947	0.079646	0.026549
Dutch	0.013889	0.250000	0.013889	**0.472222**	0.083333	0.069444	0.097222
English	0.006920	0.155709	0.128028	0.245675	**0.256055**	0.051903	0.155709

Table 3.3.: Distribution of answer types

	AG	CH	LOC	RE	TH	TMP	OW
Spanish	0.062500	0.093750	0.015625	0.125000	**0.625000**	0.078125	0
Dutch	0	0.120000	0.120000	0.040000	**0.720000**	0	0
English	0.067308	0.144231	0.096154	0.076923	**0.528846**	0.067308	0.019231

Table 3.4.: Distribution of features

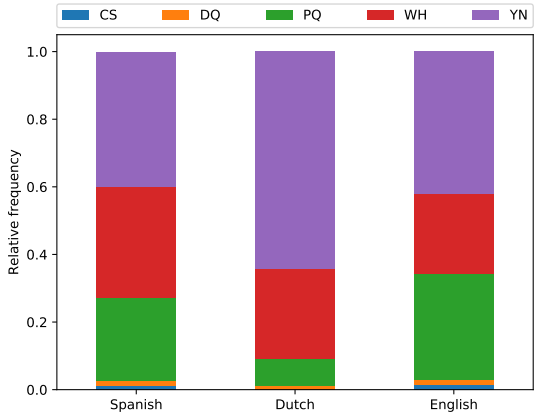


Figure 3.1.: Distribution of question types

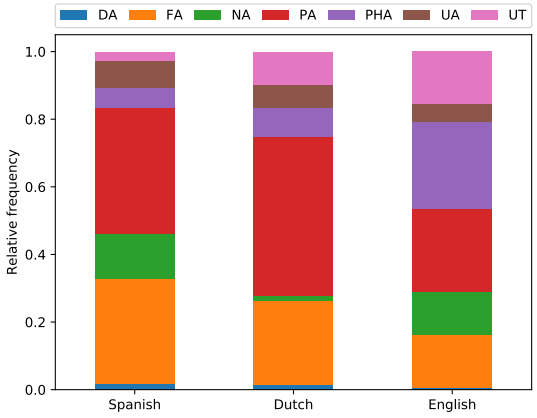


Figure 3.2.: Distribution of answer types

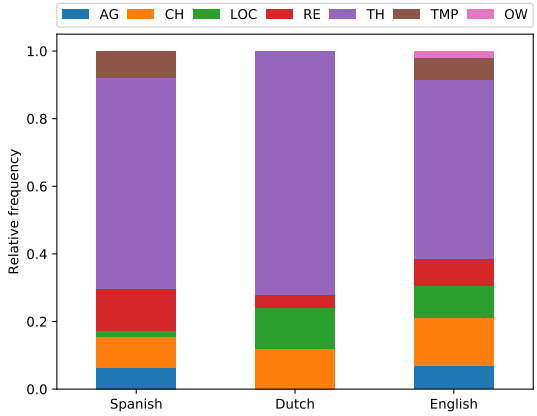


Figure 3.3.: Distribution of features

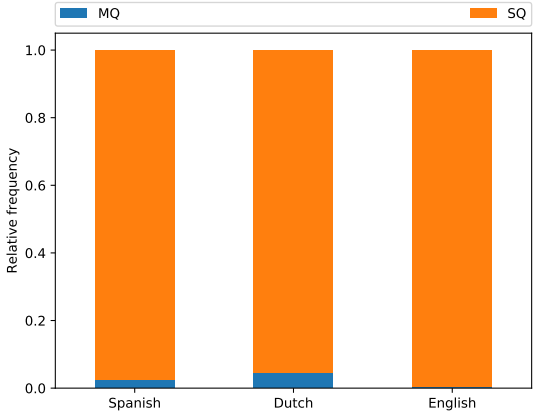


Figure 3.4.: Distribution of question complexity

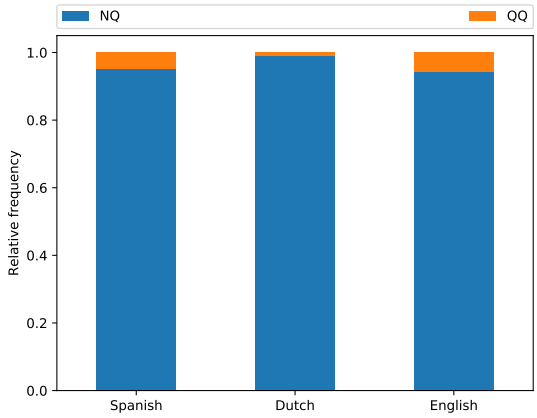


Figure 3.5.: Distribution of quoted questions

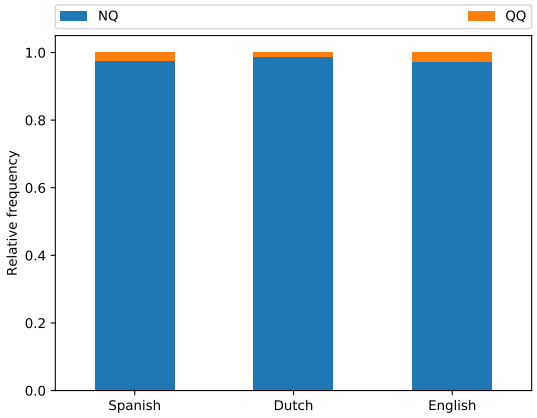


Figure 3.6.: Distribution of quoted answers

	MQ	SQ
Spanish	0.026596	**0.973404**
Dutch	0.046512	**0.953488**
English	0.004739	**0.995261**

Table 3.5.: Distribution of question complexity

	Questions		Answers	
	NQ	QQ	NQ	QQ
Spanish	**0.952128**	0.047872	**0.952128**	0.047872
Dutch	**0.988506**	0.011494	**0.988506**	0.011494
English	**0.943128**	0.056872	**0.943128**	0.056872

Table 3.6.: Distribution of quoted questions and answers

3.2.2. Agreement analysis

To evaluate the quality of the annotation guide and the annotations themselves we calculated the inter-annotator agreement of the golden corpus for the shared parts (those annotated for all the annotators). The analysis of inter-annotator agreement can give insights into the clarity of the defined procedure and the capacity of the schema to cover the different phenomena occurring in naturally spoken interactions [2]. If the annotators agree in almost all the utterances, then the procedure was understandable and clear enough to lead the annotations. If the annotators disagree on items belonging to the same category, it can be the case that the category does not represent well enough the studied phenomena.

In this section, we will present the different metrics used to calculate the inter-annotation agreement, the scores obtained and the interpretation of those values.

$\mathcal{K}$, S and Π Coefficients

Cohen's Kappa ($\mathcal{K}$) [8], S coefficient [3] and Π coefficient [27] are the most common inter-annotator agreement measures[2]. Those metrics allow us to measure the reliability of the annotations made to obtain the final version of the golden standard. They all use the concept of **observed agreement** which corresponds to the ratio of the annotations in which the annotators agree over the total of tagged utterances and the concept of chance, in which case there exists the possibility that the agreement was due to chance. It is in the definition of chance where all the three coefficients vary.

To determine the value of the scores, all of them use the formula 3.1. Where A_o corresponds to the observed agreement, and A_e corresponds to the expected agreement. The value for A_e will depend on the score that is calculated; this value is the interpretation of what conditions are considered by chance which we explained before are different for each metric. The value will be 0 if the observed agreement is the same as the expected agreement $A_o - A_e$.

$$\mathcal{K}, S, \Pi = \frac{A_o - A_e}{1 - A_e} \tag{3.1}$$

The assumptions in which each score relies on for calculating A_e correspond to the type of distribution we may obtain if all the annotators tag by chance. For S, the assumption is that the distribution will be uniform, in the case of Π the assumption is that we will obtain the same

distribution for the annotators and in $\mathcal{K}$ the assumption presumes that each annotator follows a different distribution[2].

Although the three coefficients are ratios, in the case of $\mathcal{K}$ the interpretation of the values obtained have been a source of controversy as explained in [19]. The scale of what is considered a reliable kappa value is different in different studies for instance 0.6 (moderate) and 0.8 (substantial) [15] and 0.67 (highly tentative and cautious) [14]. Consequently, It is appropriate to do an analysis which includes the three metrics and also the nature of the data.

Metrics of Golden Standard

Once we annotated the shared part of the golden standard, we extracted those annotations and proceeded with the inter-annotator agreement analysis. To do so, we implemented a script capable of doing the alignment of the annotation based on the time and text annotated and calculating the metrics for more than two annotators (in our case three annotators).

The alignment of the annotations represented a challenge because of the JSON input file (output of the post-processing step of obtaining the annotations, see section 3.1.1) order. Sometimes the annotated utterance is the same but it varies for few units for one annotator. It is because the alignment is based on the time order and that utterance will be considered different to the others even though it corresponds to the same text. For that reason, we needed a more flexible approach in which we tolerate some small differences in time with the same text that way we could distinguish if the annotations belong to the same utterance or not. Another typical case is when at least one of the annotators did not tag the utterance, but the other did, we needed to add a new category (besides the tags defined in section 2.2) that could represent this situation (the empty tag). It was also a challenge to align more than two annotators and create a general procedure capable of analyzing any number of annotators.

The first time we calculated the metrics we identified some typographical and location errors (the tag was in the wrong layer) which affected the calculations but corresponded to an error that can be avoided. For instance, if we have a mechanism in ELAN that allows establishing the possible tags for a layer, in that case, an annotator would have an error if there is a typo. The results for after the correction of those errors are shown in the table 3.7. The inter-annotator agreement was calculated pairwise and also, for all the annotators.

Layer	All				A and B				B and C				A and C			
	Avg A_o	$\mathcal{K}$	Π	S	A_o	$\mathcal{K}$	Π	S	A_o	$\mathcal{K}$	Π	S	A_o	$\mathcal{K}$	Π	S
question_type	0.73	**0.63**	0.63	0.69	0.73	**0.62**	0.62	0.67	0.7	**0.59**	0.59	0.66	0.73	**0.62**	0.62	0.69
answer_type	0.59	**0.49**	0.48	0.54	0.68	**0.6**	0.6	0.64	0.54	**0.42**	0.42	0.47	0.48	**0.35**	0.35	0.42
feature	0.9	0.67	0.67	0.88	0.92	0.74	0.74	0.9	0.89	0.64	0.64	0.87	0.89	0.61	0.61	0.86
is_quoted	0.78	0.35	0.35	0.67	0.81	0.3	0.3	0.72	0.76	0.15	0.15	0.63	0.73	0.14	0.14	0.6
complexity	0.86	0.38	0.38	0.71	0.84	-0.09	-0.09	0.68	0.84	0.2	0.2	0.68	0.87	0.26	0.26	0.74

Table 3.7.: Inter-annotation agreement scores

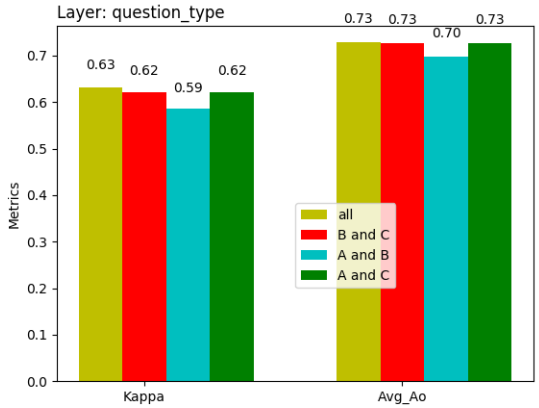


Figure 3.7.: $\mathcal{K}$ and A_o scores for questions type

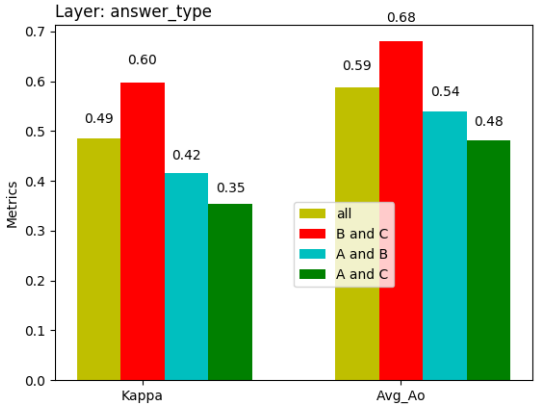


Figure 3.8.: $\mathcal{K}$ and A_o scores for answers type

In general in the table 3.7 we can see that annotator B and annotator C have a higher A_o than the pairwise with annotator A. Since that corresponds to the average of the pairwise scores, it affect the score for all the annotators. It should be noted that this annotation task is a complex one compared to other similar annotation task like POS-tagging, as there are different factors, such as semantics, syntax, prosody and span of question and answers play role in determining the required tag. As a result, it is unsurprising to obtain Kappa scores below 0.7 considering the complexity of the task.

In the case of complexity and is_quoted layers, we can see the observed agreement was substantial but the values for the metrics were considerably low. One reason for this can be that the number of possible classes for those layers is low, specifically, three (two tags from the schema and the empty tag) that makes the expected agreement A_e high, which can lead to interpretation of agreement as by chance. In the case of complexity there was no multiple question (MQ) in the part that we chose for calculation of inter-annotator agreement, and therefore the only tag used was single question (SQ) and as a result, the disagreement relies on the utterances that were not tagged (the empty tag). Because of these cases,it is crucial to support the analysis with several metrics and take into account the nature of the data.

The values obtained for the layer question_type were over 0.6 of agreement for all the annotators. This result corresponds to a moderate agreement which in turn suggests the annotation guide was clear to lead the annotators to choose the right annotations, but at the same time, it suggests that there was a need for improvement which we identified those points and addressed. We explain those improvements in the next section. These values give an insight into the coverage of the scheme. Moreover, There were exceptional cases in which was difficult to decide to which class a question belongs to, but in general, the schema was sufficient to tag all the found questions in the golden standard.

The case for the answer type is different because the schema establishes a correlation between the type of a question and the possible types for the answer to that question. Therefore, when there was disagreement in the question type, the probability of disagreement with the answer type increases and this can be seen in the table 3.7. The figures 3.7 and 3.8 show the values for $\mathcal{K}$ and the A_o for questions and answers type respectively.

3.2.3. Qualitative analysis and golden standard

After computing inter-annotator agreement scores for the common part of the 'golden' corpus, we performed a qualitative analysis of our disagreements. For each annotation for which there was disagreement, either about the span of the annotation or about one or more tags, we followed the following steps:

1. Identify the cause of the disagreement;
2. Decide what the definitive annotation should be;
3. Where possible, improve the annotation guide.

In most cases, identifying the cause of disagreement was easy. Most disagreements fell into one of four categories:

- one (or more) of the annotators made a simple mistake;
- there was an earlier disagreement that caused more disagreements;
- the annotation guide did not take a particular situation into account;
- there was a genuinely complicated problem for which it is hard to find a general solution.

Examples of simples mistakes are cases where one of the annotators overlooked a question, or where they forgot to add a particular tag (e.g. they forgot to add a feature tag for a wh-question). In such cases, it was generally obvious what the cause of the disagreement was and how it should be resolved, but it is less obvious in what way the annotation guide could be improved to prevent this kind of problem, other than to add an extra warning for annotators that it is important to check that an annotation is actually complete before moving on to the next one. A more sophisticated solution to prevent these kinds of mistakes would be to use software that checks the structure and format of annotations. Something like this does not seem to be possible in ELAN, but other methods (such as using custom XML documents and schema validation) would have allowed us to do this.

Other disagreements were hard to count them as mistakes but rather they were the logical consequence of previous disagreements. For example, suppose that one annotator has tagged a particular question with the type YN (yes/no), whereas the other two annotators believe that it is a PQ (phatic question), and suppose that the answer to this question is 'yeah'. In such a case, the first annotator, would likely interpret this answer as confirming the proposition that was contained in the putative yes/no question and tag it as a PA (positive answer); the other annotators, on the other hand, do not have this option (since positive answers are only possible for answers to certain types of questions) and will tag it as PHA (phatic answer). In such cases, the decision about the definitive annotation and about any improvements to the annotation guide depends on the decisions that were taken to address the original problem (in this case the disagreement about whether or not the question was phatic).

The third type of disagreement originated from gaps in the annotation guide. In the absence of guidance about what to do in a particular situation, the annotators often proceeded in different ways. For example, the original annotation guide did not explicitly specify whether or not it is allowed to tag a question should always be a complete utterance. This led to some annotators of tagging only a part of certain utterances as a question, while others tagged the same utterances completely as questions. To solve these kinds of problems, a (sometimes arbitrary) decision had to be make as what would be the correct annotation, and then the gap in the annotation guide could be filled accordingly.

Finally, there were problems that arose from ambiguities in the underlying linguistic data, rather than from mistakes by the annotators or gaps in the annotation guide. For example, in some cases the line between a 'real' yes/no or wh-question (where the asker of the question is interested in obtaining some piece of information) and a phatic question (where there is no information transfer involved) was very hard to draw. In these cases, we paid extra attention to original audio and tried to find prosodic cues that would help make us a decision which of these options would be the most appropriate one; although this helped in some cases, in many other cases the definitive annotation that we decided on was chosen more or less arbitrarily.

After performing the qualitative analysis, and updating the annotation guide accordingly, we would have liked to make a 'fresh start', and test the improved guide on a new corpus to see if the agreement scores would improve. Unfortunately, it was not possible to do this within the time constraints of the project.

3.2.4. Language-specific issues

When annotating the two non-English corpora, we came across several issues which prevented us from directly applying the annotation guide, which was created based on the English corpus (SCoSE), to these corpora. Some of these problems have to do with the languages themselves, while others are due to transcription errors and different transcription methods. In this subsection, we summarize these problems and explain how we addressed them (where possible).

Spanish

- Some utterances that are not questions are marked with a question mark in their transcription, and some utterances that clearly are questions lack a question mark. For consistency reasons, we decided to ignore these utterances and leave them unannotated.
- In written Spanish, questions are not only marked by a question mark at the end of the question, but also by an inverted question mark at the beginning of the question. The transcriptions did not include these inverted initial question marks, which sometimes caused ambiguity about the start of the question.
- Spanish wh-words are distinguished from demonstrative pronouns by the use of an accent. For example, *qué*, with an accent, means 'What', whereas *que*, without accent, means 'that'. However, in the transcriptions, these accents are sometimes mistakenly omitted. While this is unlikely to be a problem for human annotators, it might have a negative impact on the performance of automated tagging systems (especially if these rely on heuristics such as the presence of a wh-word).
- In the Spanish corpus, overlaps between speech turns (i.e., both speakers are talking at the same time) are much more frequent than in the English corpus. This causes a problem in cases where both speakers ask a question at the same time, since, due to the way that we had configured ELAN, we could only assign one question annotation to a given time span. We solved this issue by dividing the time span of the two questions into two parts, and then assigning the annotation of one question into first part and the annotation of the other question to the second part. The obvious disadvantage of this approach is that neither of the two annotations will have the correct time span; however, our post-processing script automatically corrects this issue.
- Many 'utterances' in the transcription actually comprise of multiple utterances. This meant that we could not rely on utterance boundaries for determining the end and starts of questions, but had to find the correct time spans manually.

Dutch

- Like in the Spanish corpus, the Dutch corpus contains a large number of overlaps between speech turns. However, there were no overlaps between questions, so this did not cause any problems.
- For each dialogue, the CGN offers both a version that is segmented utterance-by-utterance, and one that is segmented word-by-word. Initially, our plan (for all corpora) was to manually find the time spans of each question and answer, instead of relying on the utterance boundaries as defined in the transcriptions. Under this approach, it would be helpful to know where the word boundaries are, and so we decided to use the word-by-word transcriptions. However, later we decided to rely on utterance boundaries to determine question and answer spans after all, but since, by this time, the Dutch corpus was already completely tagged, and there was not enough time to re-annotate the corpus using the utterance-by-utterance transcriptions, so we decided to keep these annotations as they were.

4. Machine learning

4.1. Machine learning experiments

One aim of this project was to explore machine learning algorithms to automate the annotation process. Considering the nature of the task and the amount of data we had available, we decided to take two approaches into account. The first approach was to use a traditional statistical classification algorithm; and the second one was to use more recently used neural networks. For the moment the automatic annotation process only classifies questions. There are some reasons for which we did not include the answers for this part. Firstly, answers are more complicated to be detected. The span of the answers are varying, and more than one utterance can be considered as the answer of a question. Moreover, for some questions there are no answers. These difficulties make the automatic detection and classification of the answers an arduous task. As a result, we decided to focus on question type identification. In the following subsections, we explain the algorithms we used, provide technical details of each, and finally, we give the results and their analysis.

4.1.1. Classical statistical approaches

Decision trees

Before the emergence of modern neural network approaches, decision trees were a commonly used algorithm in NLP research and where known for producing good results for classification tasks. Domains where this algorithm has been successfully applied include coreference resolution and for semantic role argument labeling [6, 29]. Decision trees, among other algorithms, have also been successfully used for question classification tasks [34]; however, this task was significantly different from the one that we are presenting here it is mostly focused on classifying features of wh-questions, rather than classifying questions and their discourse contribution in general.

However, the main reason for using decision trees is that we believe that they are a logical extension of our previous work on writing an annotation guide, and that they are a natural fit to our problem. In our annotation guide, we provide step-by-step instructions for human annotators on which cues in the data to look for and how to use these to assign the correct question tag. A decision tree algorithm works in much the same way: when provided with a sample, represented set of features (the cues), it follows a sequence of decision steps in order to classify the sample by asking questions about each feature. However, decision trees have the advantage of being much more general: rather than relying on a predefined sequence of steps to take, the optimal sequence is learned based on training data.

This algorithm is a supervised learning algorithm which places the best attribute of the dataset at the root value and splits the training set into subsets according to a metric like information gain. Information gain is the amount of information that a feature provides about a class. The attributes that have higher information gain reduce the amount of entropy and split the data optimally.These steps are continued until leaf nodes are produced for all the branches of the tree. For classifying a new instance, the value of the feature of the instance is checked against the value of the nodes and this process continues until it reaches a leaf node which predicts the class of the new instance.

Feature extraction

A crucial aspect of classical classification approaches is the extraction of relevant features for the problem at hand. These features should be extracted in a way that, on one hand, does not make the other features redundant and at the same time, does not lead to over-fitting problems [31]. The list of the features and their corresponding values are shown in table 4.1.

Feature	Description	Value
has_wh	The question contains a wh-constituent	True, False
has_or	The question has contains the word ”or”	True, False
has_inversion	The question has inverted structure	True, False
has_tag	The question is a tag question	True, False
has_utt_similar	The last utterance has at least 50% similar words	True, False
last_utt_incomplete	The last utterance is incomplete	True, False
has_cliche	The question contains a cliché (e.g. ”really?”, “you know?”)	True, False
length	The length of the question (number of words)	Numerical value

Table 4.1.: Extracted features for the classification task

In choosing our feature set for the task of classification, we looked at our schema and chose the attributes that either distinguish or differentiate questions and can be easily recognized by a computer program. Some of the features like *has_wh*, *has_or* and *has_inversion* are the main clues in recognizing some types of the questions. Some of the features like *has_utt_similar* and *last_utt_incomplete* are not inherently a defining factor to determine types of the questions, but they increase the probability of some types of the questions like completion suggestion. For example, in (13), *last_utt_incomplete* is a cue for determining that a question is a completion suggestion question, but having a positive value for *last_utt_incomplete* is not, in itself, a sufficient or necessary condition for being a completion suggestion.

(13) A: In fact we’ve kind of looked just ...
B: just for the hell of it? (SCoSE/Amy, 640-641)

Among these features, the *length* of the question is the only feature that has a numerical value and is not part of the annotation procedure. This feature is not necessarily a defining characteristic for any of the question types, but our intuition is that usually, phatic question are shorter than the other types of the question and hence, that this feature can be helpful for the classifier to capture this type of the question.

After deciding on the set of features and their values, we used the Python library *sklearn* [21] to build up the decision tree model, train it and assess its feature importances (i.e., the influence of each feature on the decision process). The result is shown in figure 4.1. The result for the performance of the model can be seen in section 4.2.1.

The scikit-learn package assesses the features based on the Gini importance. The Gini importance tells us how many times a feature has been used to split a node in the tree. The result is normalized and presented on scale of 0 to 1. According to the results, the *lenght* has the highest Gini score and the features *last_utt_incomplete* and *last_utt_similar* have gotten zero Gini score. The strategy that a machine can use to obtain a question type is different from a human annotator. First, the *length* is a feature that a computer can use as an important factor to determine the type of the question. Second, the features that can be perceived as a good clue for a human annotator for differentiate types of questions are not used by the algorithm the case of the features related to the last utterance. However, the nature of the training data, like the distribution of question types may affect the feature importance.

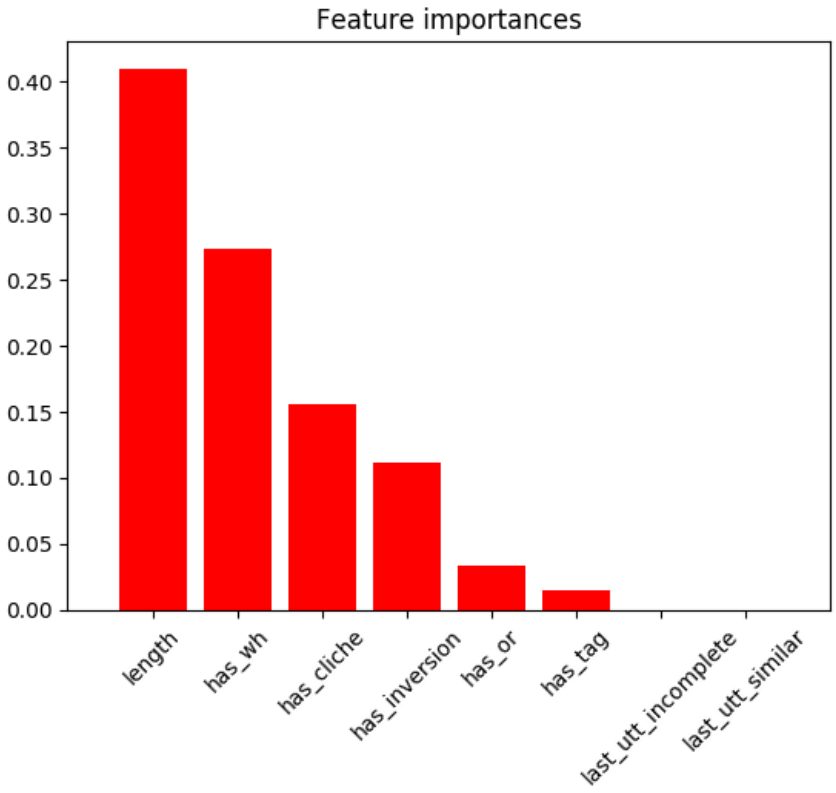


Figure 4.1.: Informative features

4.1.2. Neural network approaches

In recent years, artificial neural network (ANN) approaches to machine learning have become more and more popular in NLP and in artificial intelligence more generally (see [17] for a discussion of these developments and their impact on computational linguistics). Neural networks can have many different types of architectures and sizes. We experimented with three different models, all of them very small-scale (in terms of the number of layers and the number of neurons per layer), but with somewhat different architectures.

One of the most important problems when designing a neural model is choosing how to represent the input data; this choice significantly impacts the learning process and determines what kind of architecture is needed. Each of our three models represents question in a different way. First of all, our baseline model uses feature vectors; this both simplifies the problem and enables us to better compare the neural models to the decision tree algorithm. Our second model uses a 'bag of words' (BOW) representation, in which a sentence is represented in terms of its word counts. Under this approach, information about the syntactic and semantic structure of the questions is lost. Trying to use a model like this is an interesting test for exactly to what extent it is possible to classify questions without such information. Although *a priori* this might seem an impossible task, our decision tree experiments showed that sentence length and the presence of wh-words and cliches are useful features for classifying questions, information that is also (implicitly) present in BOW representations. Finally, we use 'raw' sentences, represented as a sequence of word representations, as input for our third model, which will be a test for how well this model is at extracting useful features (possibly including syntactic and semantic ones) on its own.

Baseline: Multi-Layer Perceptron

The first model that we used is the Multi-Layer Perceptron (MLP) classifier from the sklearn package. The reason for choosing this model is that it is simple but well-established, and could be trained and tested using the same interface as the decision tree algorithm (which was also taken from the sklearn package). The MLP takes the same feature vectors that were used for the decision tree algorithm (see above) as inputs and then feeds these through a single hidden layer, before predicting the question type.

Prototype: BOW classifier

Unlike our baseline model, which was already published in a ready-to-use form, our second model did not previously exist as such. It is based on a simple model for language classification that was published as part of a tutorial aimed at beginning users of a popular deep learning framework (PyTorch) [10]. This model is a simple single-layer perceptron that takes as input a bag-of-words vector and maps it to a vector of scores for each of the class labels. As a final step, the softmax function is applied to these scores to produce a probability distribution.[1] We adapted this model to our dataset and extended it with a ReLU function, which is applied after the single-layer perceptron but before the softmax layer.[2] The structure of the model is summarized in figure 4.2.

We think that this model is interesting for two reasons. Firstly , as mentioned above, it is a test for representing sentences in a 'poor' way (i.e. without structural information). Moreover, the model is very simple: it has a single linear layer and a limited number of parameters to learn. For this reason, it seems quite suitable for training on a small dataset.

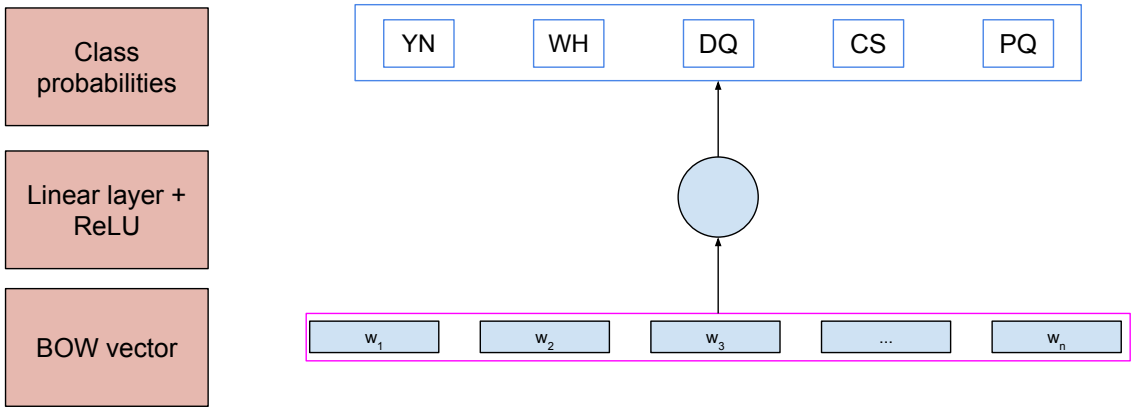


Figure 4.2.: Architecture of our BOW prototype

RNN prototype

Our final model is the one that we think is potentially the most interesting one. One of the types of artificial neural network models that has become especially common in NLP are recurrent neural networks (RNNs). The main advantage of such networks is that they are able to

[1] The softmax function, defined by $f(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$ [23], is a commonly used component in neural networks, and maps any sequence of numbers to a sequence of numbers between zero and one whose sum is equal to one.

[2] The ReLU function function is defined by $f(x) = \max(x, 0)$ [22]. Although it is very simple, it is also powerful because it introduces a non-linear relationship between the input and the output (its graph is not a straight line), and enables our model to approximate a wider set of functions.

detect relationships between elements in a sequence, which is crucial for modeling the syntax and semantics of natural language. One of the problem domains in which RNNs have been successfully used is machine translation, using so-called sequence-to-sequence (seq2seq) models. A seq2seq model consists of two main components: an encoder and a decoder, both of them RNNs. In such models, sentences in the input language are fed, word by word, to the encoder. The result of this is an intermediate representation, that, when training is successful, will contain information about the meaning and syntactic structure of the input sentence. The decoder component then takes this representation as input and produces a sentence in the output language.

We believe that, at least in theory, RNN-based models could be useful for our question classification task, since this tasks relies heavily on syntactic and semantic information. However, a crucial difference between our task and other tasks where RNNs are commonly used, such as machine translation, is that we are not interested in mapping a sequence to another sequence, but in mapping a sequence to class labels. Hence, seq2seq models like those used for machine translation and related domains are not suitable for our task. However, it might be possible to adapt the seq2seq architecture to our task by replacing the decoder component by a neural network that takes the intermediate representation produced by the encoder as input and outputs a class label. Such an approach has already been successfully adapted in related domains, for example in document sentiment analysis [30].

Designing a neural network from scratch is not a trivial task, and any serious attempt to construct a model that would produce usable results would be beyond the scope of this project. Thus, the model that we present here is no more than a proof of concept. The model is based on a simplistic seq2seq model for machine translation that was published as part of a tutorial aimed at beginning users of PyTorch [26]. We modified this model by removing the RNN decoder and replacing it by a simple linear classifier that maps the encoder input to a probability distribution of the five question type classes.[3]

The structure of our model is shown in figure 4.3. The initial inputs to the model are individual words encoded in so-called *embeddings*, which are vectors that, as the model is trained, accumulate contextual information about the words that they represent. These embeddings are fed to a Gated Recurrent Unit (GRU) network, which is a commonly used variation of RNNs [7]. The output of this layer of the network is a sequence of 'hidden states', which we concatenate to obtain a single representation of the meaning of the sentence. This vector is then fed as input to the linear classifier, which produces a score for each of the five possible question types. Finally, by using the softmax function, we obtain a probability distribution.

We performed two different experiments using this setup: in the first one, the initial values of the embeddings are random and have to be learned by the model. In the second experiment, we used pre-trained embeddings that were published as part of the *spaCy*[4] Python library [12]. By making sure that the model can make use of contextual information (which in turn is a proxy for syntax and semantics) from the very beginning, the amount of learning work that the model has to do is reduced. Using pre-trained vectors is a common strategy in experiments with small training datasets [25].

The model was trained on our annotations of the Amy dialog in the SCoSE corpus. The maximum sequence length was set to 20. This number was found by experimentation and was chosen because it finds a good balance between covering as much of the training data as

[3] A linear classifier is a model of the form $f(x) = Wx + b$, where the parameters W and b are a weight matrix and a bias vector, respectively. These parameters are adjusted during training in order to optimize for the desired outputs of $f(x)$. Note that a perceptron, the simplest form of a neural network, is essentially a linear classifier combined with a non-linear activation function which enables it to approximate a wider range of mathematical functions.

[4] spaCy contains several neural models; the model that we used to get our embeddings is called *section-en_core_web_sm*; for more details see https://spacy.io/models/en#section-en_core_web_sm.

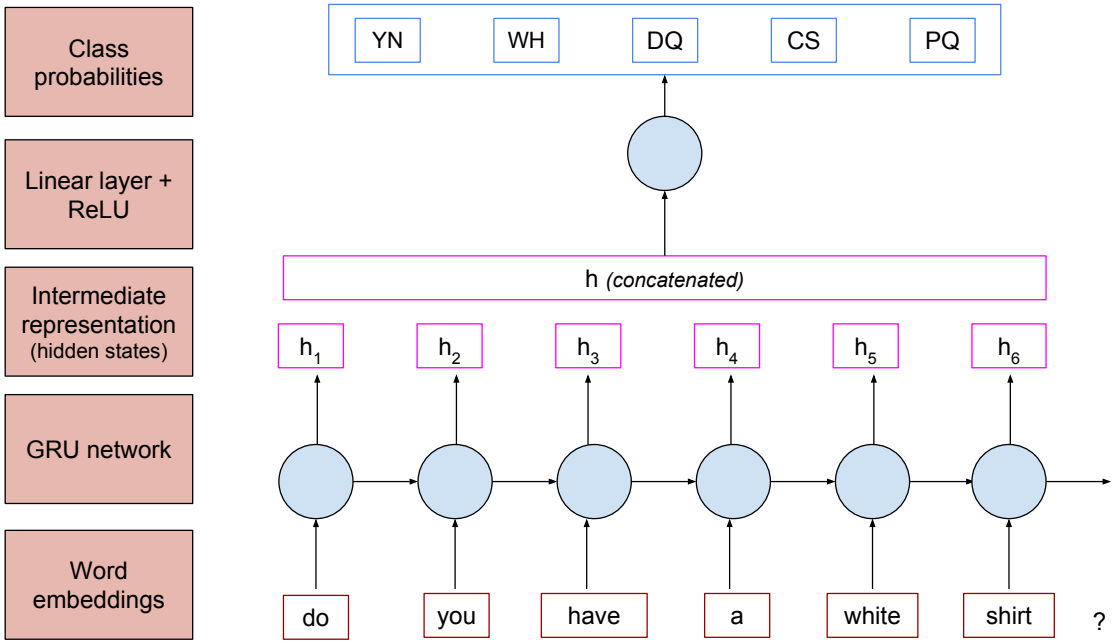


Figure 4.3.: Architecture of our RNN prototype

possible and keeping the limit low to keep the model as simple and small as possible. Other (hyper)parameters of the model, such as the learning rate ($l_r = 0.01$) were taken directly from the original model that our work is based on. The number of units in the hidden layer of the GRU network (300 hidden units) followed directly from the pre-trained embeddings that we used; for consistency reasons, we used the same hidden size in both versions of the experiment.

4.2. Machine learning results and evaluation

In this section we describe the experiments carried out for the decision tree model and the RNN prototype and their results for the prediction of training and testing sets. For both classifiers we evaluated their performance with the standard metrics: accuracy, precision, recall and F1 measure.

4.2.1. Decision tree

After the extraction of features which explained in the previous section, we performed some experiments to obtain a decision tree model using the golden standard corpus as the training data and the experimental corpus as the testing data. For both datasets, we used labeled annotations for evaluation. In the following, we will explain the procedure to build up the model and discuss the results for the evaluation metrics.

To calculate the decision tree model, we used an existing implementation available in sklearn. To do so, we needed to pre-process the output of the feature extraction procedure to build the appropriate input for sklearn, in which we represented a question as a feature vector. Except for the length feature, which is represented by an integer value, all features are binary and are represented by the integer 1 (true) or 0 (false).

Once the new data structures were defined, we passed them to sklearn to build up the model and as mentioned above, the model was trained using the golden standard corpus and tested

on the experimental corpus. The result of this process was a decision tree which a graphical representation of it can be found in appendix A.

We evaluated the performance of this model by testing it on both seen data (i.e. the same data that it was trained on) and unseen data (the test corpus). The metrics used for this evaluation are accuracy, precision, recall and the F1 measure. Accuracy simply measures how many annotations of the total were correctly predicted. However, this measure needs to be interpreted with care: it may, in some cases, produce high scores for models that have no predictive value at all (for example, models that assign the same category to all samples). Hence, it is imperative to calculate other metrics. The evaluation should also include precision and recall values. Precision refers the number of correct annotations within the set annotations classified as positive. On the other hand, recall refers to the number of correct annotations within the complete set of positive annotations (true positives and true negatives). Finally, the F1 measure represents a balance between recall and precision and evaluates how well our model performs regarding precision and recall at the same time. Since the number of classes is more than two, the calculation for the metrics requires an especial treatment. Particularly, we used the method *macro average* of precision, recall and F1 measure. This method, calculate the metrics for each class (for instance, the precision for *YN*, the rest of the tags will considered as negative cases) and performs the regular average over the total number of classes. The macro average allows to have an overview of the model over the whole data, which is of our interest.

We ran several experiments to look over the impact of different configurations for training and testing datasets and the features fed to the decision tree. A summary of the scores obtained for the different configurations is shown in the table 4.2. The main configuration uses the golden standard corpus as the training set and the experimental corpus as the testing set (see section 3.1.3). To confirm the importance of the contributions of all the annotators in the capability of the model in prediction, we interchanged the training and testing sets, so the experimental corpus became the training set and the golden standard as the testing set (**experiment 1**). To determine if the scores obtained for the testing set corresponded to the fact that the annotations for the experimental corpus were done for only one of the annotators we ran the **experiment 2** using only the contribution of the annotator A in the golden standard as the training set. In the figure 4.1 we can see the order of importance of the features for the model (main configuration). We ran two experiments to verify if the most important features were enough to have a good model of classification. These two experiments have the main configuration for training and testing sets. We only fed the length (**Feature:Length**) and recalculated the metrics and in the second experiment we fed only the length and has_wh features to the decision tree and recalculated the metrics again.

The table 4.2 shows the different values of the metrics for the training and testing sets. As expected, the values for the training sets were always greater than the values for the testing sets in all the configurations. The best model in terms of F1 measure for unseen data was the main configuration, 0.58. However, the model for experiment 1 also was able to learn to identify different types of questions. The difference with respect to the main configuration is only 0.08 for the F1 measure. The size of the training set for the main configuration is 238 and for the configuration of experiment 1 is 116. Although there was a difference in size, the model presented a similar performance to the one for the main configuration. This explains how the performance for the experimental corpus is related to the nature of the training set and the distribution of question types. In figures 4.5 and 4.4 we can see that the distribution is different for both corpora. Most of the questions in the experimental corpus were *YN* about 0.5 and then after that, the second most common type was *WH* approximately 0.35 while for the golden standard corpus the most common types were *PQ* and *YN* with about 0.45 and 0.38 respectively. Also, We believe that the diversity of the training set is relevant to build up a model able to generalize. On the other hand, in the experiment 1, we can see an overfitted

model that presents a good performance for the experimental corpus but a poor performance for the golden standard.

The results for the experiments concerning the features show the accuracy paradox [33] where the accuracies are considerably good 0.68 and 0.78 for the first experiment (only the length and the second experiment, only the length and has_wh respectively), but the rest of the metrics are substantially low(below of 0.50 for the training set and below 0.40 for the testing set). These two configurations do not have a predictive power, even though, the models are based on the features with more importance according to 4.1. In spite of the importance of the *length* feature for the computational approach, the decision tree still needs the other features to build up a model capable of learning the underline characteristics needed to identify the different question types.

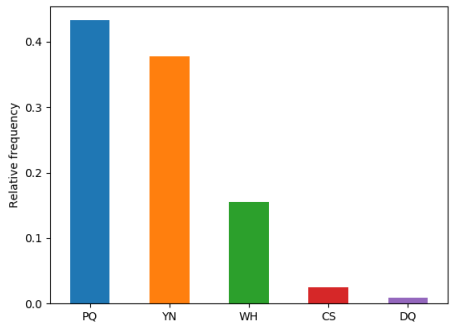


Figure 4.4.: Distribution of question type for the golden standard corpus

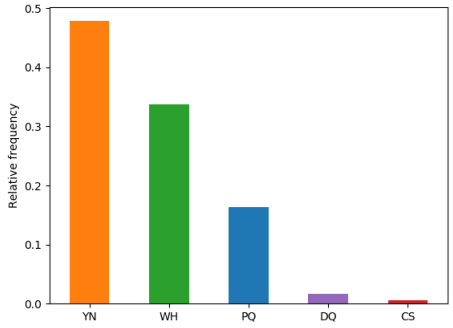


Figure 4.5.: Distribution of question type for the experimental corpus

Although the model was able to learn characteristics of the different questions types to some extent, given that the F1 measure for unseen data is only 0.58, it is not sufficient to conclude that our model learned to generalize. One possible next step is to collect more data for training and testing sets which may improve the performance of the classifier. As discussed in section 4.1.1, the features play an important role in a decision tree; hence, to improve the performance, it is possible to revise the feature set and explore new features that may not be part of the process described in the annotation guide. Such features can be based on syntactic information or on n-grams [34].

4.2.2. Neural network models

Our neural models were trained by feeding all of the input sentences in the golden standard corpus (one at a time) to the model several times (or 'epochs'). For the first model, the MLP from the sklearn package, the number of epochs was determined automatically. For the other two models, this was not possible; instead, we trained them for 100 epochs and evaluated the results after each epoch. From this analysis, we learned that in all cases, the model had converged after around 70 epochs.

A problem during testing of the RNN model was that there were case in which our model was not able to assign a question type to some of the questions, either because the question exceeded the sequence length limit of the model, or because it contained words that were not present in the training data. In these cases, we defaulted to assigning the question type YN based on the question type distribution found for English (see section 3.2.3). However, this happened only three times and is unlikely to have had any effect on the learning process.

For each of the neural models, we computed the same evaluation metrics as for the decision trees (see the previous subsection for an explanation of these metrics). Our results are summarized in table 4.3. As can be seen in the table, the results are mixed: while both the BOW

Data set	Accuracy	Precision	Recall	F1
Main configuration				
Golden Standard (Training)	0.80	0.90	0.71	**0.74**
Experimental (Testing)	0.73	0.56	0.60	**0.58**
Configurations changing training and testing sets				
Experiment 1				
Experimental (Training)	0.89	0.94	0.89	0.90
Golden Standard (Testing)	0.62	0.40	0.41	0.39
Experiment 2				
Golden Standard (Annotator A's part - Training)	0.88	0.74	0.72	0.73
Experimental (Testing)	0.65	0.52	0.51	0.50
Configurations changing the features				
Feature: Length				
Golden Standard (Training)	0.68	0.48	0.37	0.36
Experimental (Testing)	0.52	0.22	0.29	0.24
Features: Length and Has_WH				
Golden Standard (Training)	0.78	0.49	0.47	0.48
Experimental (Testing)	0.70	0.40	0.40	0.39

Table 4.2.: Scores for the different decision tree configurations. Main configuration corresponds to the model using the golden standard corpus for training. After the configurations were the training set and the testing set were changed to analysis the contribution of the three annotators and annotator A. Finally, the configurations where the set of features used to build up the model were changed.

Data set	Accuracy	Precision	Recall	F1
Perceptron (feature-based)				
Golden Standard (Training)	0.63	0.39	0.39	0.37
Experimental (Testing)	0.48	0.37	0.34	0.30
BOW Classifier				
Golden Standard (Training)	0.87	0.53	0.51	0.51
Experimental (Testing)	0.66	0.39	0.35	0.36
RNN Classifier				
Golden Standard (Training)	0.92	0.56	0.57	0.56
Experimental (Testing)	0.40	0.17	0.18	0.16
RNN Classifier (pre-trained embeddings)				
Golden Standard (Training)	0.90	0.54	0.56	0.55
Experimental (Testing)	0.43	0.12	0.18	0.13

Table 4.3.: Scores for the different neural models

classifier and the two versions of the RNN classifier have a very high accuracy score on the training set, their precision and recall scores are far behind that of the decision tree algorithm. Moreover, all of the neural models perform badly, at least compared to the decision trees, on unseen data; the only model that comes somewhat close is the BOW classifier. Finally, our hypothesis that supplying our RNN models with pre-trained vectors would make learning easier is disproved by the results: there is little to no difference between the two versions of the model. However, it is possible that with more data, or with better pre-trained embeddings, we would have seen a difference.

All in all, we can say that the performance of the neural models is not as good as that of the decision trees. This was only to be expected, given that in general, neural networks need much more training data than we had available. However, the models still perform well above chance level, at least on accuracy (given that there are five different classes, a completely random algorithm would be correct around 20% of the time). Moreover, we believe that, with more training data and a more sophisticated architecture, the results could improve dramatically.

5. Conclusions

In this report, we presented the results of our work on our supervised project. We had three main objectives: (1) designing a classification schema for questions and answers, (2) writing an annotation guide and using it for manual annotation, and (3) exploring machine learning techniques for automating this process. All of these tasks were completed, although, as always, there is much room for improvement. In this final chapter, we start by looking back at our work and identifying areas that could be improved (section 5.1). After this, in section 5.2, we explore how our work could be built upon in the future.

5.1. Introspection

In this section, we review our own work on each of the tasks that we completed. For each task, we identify problems and limitations, and make suggestions as to how these might be overcome.

5.1.1. Annotation schema

As previously noted, when designing our annotation schema we needed find a balance between, on one hand, trying to be as precise and fine-grained as possible, and, on the other hand, keeping the schema as simple as possible to annotate by both human and automatic annotators. We believe that we have been reasonably successful in finding such a balance; however, there are some subtleties that a discourse modeling system would need to be aware of but that our schema fails to address. For example, we annotate an answer as 'UA' (uncertain answer) if the answerer does not give an unambiguous answer or indicates that they are uncertain of the correct answer. However, this approach does not take into account that some uncertainties have a very different semantic and pragmatic meaning than others. For example, consider the fragment in (14):

(14) A: What does he want to do? *[Question type: WH, Feature: TH]*
B: He doesn't know exactly. *[Answer type: UA]* (SCoSE/Amy, 3562-3563)

Since the answer expresses uncertainty, it was tagged as UA. This might seem quite reasonable, but, suppose that the answer would have been 'I don't know exactly', the same tag would have been applied, even though that answer would have expressed a fundamentally different kind of uncertainty. In fact, the actual answer does not express any uncertainty on the part of the speaker, but under our current schema there is no better way of classifying it.

5.1.2. Annotation guide

Although the guide has been developed mainly based on English language since many of the definitions and instructions make reference of specific features of the English language, the experiments with the corpora of Dutch and Spanish languages showed that the instructions of the annotation guide can be adapted to other languages. Moreover, the guide also relies on some features that are specific to the SCoSE corpus, in particular to the way in which utterance

boundaries are indicated. However, there are differences in the way that different corpora implement these conventions, especially when it comes to marking utterance boundaries. Hence, if we were to extend our annotation guide to other corpora, some changes would need to be made to the annotation procedure.

A different problem is that, despite our efforts to write the annotation guide as explicitly and precisely as possible, there are still a number of places in the guide where the annotator is asked to use their 'semantic judgment', we would like to remove such instructions to make the guide more objective. Given that there are many linguistic phenomena that are not yet understood from a scientific point of view, but about which humans are able to make intuitive judgments, in some cases, it is likely not possible to replace these instructions by more rigorous ones. However, in other cases, we might have tried to find a more principled solution, for example by making reference to the linguistic literature about the topic at hand.

5.1.3. Annotation experiments

As we showed in section 3.2.2, the inter-annotator agreement is in general not very high. However, it is not completely clear what exactly the causes of this were and how much each cause affected the scores. In our qualitative disagreement analysis (see section 3.2.3), we identified several causes of disagreements, and updated the annotation guide to try to address these. Unfortunately, we did not have time to use the improved annotation guide to annotate a new corpus and test if and how much this would increase the agreement scores. Hence, we do not know to what extent our low agreement scores were caused by gaps in the annotation guidelines, or to what extent other factors (e.g. the inherent difficulty or subjectiveness of the task) might have played a role.

A related problem stems from the fact that all of the annotators were also group members (and vice versa) and were also involved in designing the classification schema and writing the annotation guide. This might have affected the annotations, and also the agreement scores, in several ways. On one hand, it is possible that, because of the fact that during the annotation process, we already had a clear conceptual understanding of our the classification schema and knew what cues to look for when annotating, and therefore produced better annotations than an outside annotator would be able to do based on the annotation guide. Hence, it is possible that there are further weaknesses in the annotation guide that we did not identify because our prior knowledge of the classification schema prevented us from making certain mistakes. On the other hand, it is also possible that precisely this prior knowledge of the task led us (or some of us) to be biased in favor of our own conceptual understanding of the classification scheme rather than solely relying on the annotation guide.

In any case, it would have been very interesting to have one or more people outside of our project group annotate (part of) our corpus based on the annotation guide, and to see if any difficulties would come up and to what extent their annotations would differ from our own. Furthermore, asking other native speakers of Spanish and Dutch to annotate part of the corpora in these languages would have enabled us to calculate agreement scores for those corpora. Although we have started preparing such an experiment, due to time-related and practical concerns we were not able to actually conduct it yet.

5.1.4. Machine learning

Regarding our machine learning experiments, the main problem is that we did not have enough annotations to train the models on. This not only affected the quality of the results, but the analysis of the results as well, because it is not clear how a machine learning algorithm would have performed with more and diverse data. Particularly, we can not predict if the decision tree will still have a better performance than the neural network approaches.

A potentially interesting way to work around this problem would have been to use unsupervised learning techniques. The obvious advantage of unsupervised approaches is that they do not require labeled data, but it is unclear whether or not such approaches would be useful for our purposes. This has several reasons: firstly, in unsupervised algorithms it is much harder to control what kind of classes the algorithm will make, and it is extremely unlikely that such an algorithm would be able to create classes that correspond exactly to our annotation schema. Secondly, it is unclear how the results of such an algorithm could be evaluated. However, it might still have been interesting to further explore whether such a approach might have been feasible.

WARNING: this section is a stub. It will be expanded in a later version of this report.

5.2. Future work

As it has been said in this report, this project can be thought as an early attempt for annotation of discourse models, and as a result, there could be potentially many ways to continue and improve what we have done for this project.

First of all, **WARNING:** this section is empty for now. It will be added in a later version of this report.

Bibliography

[1] Maxime Amblard, Karën Fort, Caroline Demily, Nicolas Franck, and Michel Musiol. "Analyse lexicale outillée de la parole transcrite de patients schizophrènes". In: *Traitement Automatique des Languages* 55 (3 2015), pp. 91–115.

[2] Ron Artstein and Massimo Poesio. "Inter-coder agreement for computational linguistics". In: *Computational Linguistics* 34.4 (2008), pp. 555–596.

[3] E. M. Bennett, R. Alpert, and A. C. Goldstein. "Communications Through Limited-Response Questioning*". In: *Public Opinion Quarterly* 18.3 (1954), pp. 303–308. DOI: 10.1086/266520. eprint: /oup/backfile/content_public/journal/poq/18/3/10.1086/266520/2/18-3-303.pdf. URL: http://dx.doi.org/10.1086/266520.

[4] Maria Boritchev. "Approaching dialog modeling in a dynamic framework". MA thesis. Université de Lorraine, 2017.

[5] Alexandra Canavan and George Zipperlen. *CALLFRIEND Spanish-Non-Caribbean Dialect LDC96S58. Web Download. Philadelphia: Linguistic Data Consortium*. 1996.

[6] John Chen and Owen Rambow. "Use of deep linguistic features for the recognition and labeling of semantic arguments". In: *Proceedings of the 2003 conference on Empirical methods in natural language processing*. Association for Computational Linguistics. 2003, pp. 41–48.

[7] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation". In: *CoRR* abs/1406.1078 (2014). arXiv: 1406.1078. URL: http://arxiv.org/abs/1406.1078.

[8] Jacob Cohen. "A Coefficient of Agreement for Nominal Scales". In: *Educational and Psychological Measurement* 20.1 (1960), pp. 37–46. DOI: 10.1177/001316446002000104. eprint: https://doi.org/10.1177/001316446002000104. URL: https://doi.org/10.1177/001316446002000104.

[9] Philippe De Groote. "Towards a montagovian account of dynamics". In: *Proceedings of Semantics and Linguistic Theory XVI*. 2006.

[10] *Deep Learning with PyTorch. Deep Learning Building Blocks: Affine maps, non-linearities and objectives*. 2017. URL: https://pytorch.org/tutorials/beginner/nlp/deep_learning_tutorial.html (visited on 05/28/2018).

[11] English Linguistics, Department of English at Saarland University. *SCoSE Part 1: Complete Conversations*. 2017. URL: http://web.archive.org/web/20080207010024/http://www.808multimedia.com/winnt/kernel.htm.

[12] Explosive AI.

[13] Alice F. Freed. "The form and function of questions in informal dyadic conversation". In: *Journal of Pragmatics* 21.6 (1994), pp. 621 –644. ISSN: 0378-2166. DOI: https://doi.org/10.1016/0378-2166(94)90101-5. URL: http://www.sciencedirect.com/science/article/pii/0378216694901015.

[14] K. Krippendorff. *Content Analysis: An Introduction to Its Methodology*. Sage commtext series. Sage, 1980. Chap. 12. Reliability. URL: https://books.google.fr/books?id=WlXzXwAACAAJ.

[15] J. Richard Landis and Gary G. Koch. "The Measurement of Observer Agreement for Categorical Data". In: *Biometrics* 33.1 (1977), pp. 159–174. ISSN: 0006341X, 15410420. URL: http://www.jstor.org/stable/2529310.

[16] Bronisław Malinowski. "The problem of meaning in primitive languages". In: *The meaning of meaning, Supplement I*. Ed. by Charles Kay Ogden and I.A. Richards. London: Kegan Paul, 1936, pp. 296–336.

[17] Christopher Manning. "Computational linguistics and Deep Learning". In: *Computational Linguistics* 41 (4 2015), pp. 701–707.

[18] Max Planck Institute for Psycholinguistics, The Language Archive, Nijmegen, The Netherlands. *ELAN (version 5.0.0-beta)*. 2017. URL: https://tla.mpi.nl/tools/tla-tools/elan/.

[19] Banerjee Mousumi, Capozzoli Michelle, McSweeney Laura, and Sinha Debajyoti. "Beyond kappa: A review of interrater agreement measures". In: *Canadian Journal of Statistics* 27 (1), pp. 3–23. DOI: 10.2307/3315487. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.2307/3315487. URL: https://onlinelibrary.wiley.com/doi/abs/10.2307/3315487.

[20] Nelleke Oostdijk. "The design of the Spoken Dutch Corpus". In: (Nov. 2001), pp. 105–112.

[21] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[22] *PyTorch documentation: ReLU*. 2018. URL: https://pytorch.org/docs/master/nn.html#relu (visited on 05/28/2018).

[23] *PyTorch documentation: Softmax*. 2018. URL: https://pytorch.org/docs/master/nn.html#softmax (visited on 05/24/2018).

[24] Manuel Rebuschi, Maxime Amblard, and Michel Musiol. "Using SDRT to analyze pathological conversations. Logicality, rationality and pragmatic deviances". In: *Interdisciplinary Works in Logic, Epistemology, Psychology and Linguistics: Dialogue, Rationality, and Formalism*. Ed. by Manuel Rebuschi, Martine Batt, Gerhard Heinzmann, Franck Lihoreau, Michel Musiol, and Alain Trognon. Vol. 3. Berlin/Heidelberg: Springer, 2013.

[25] Seyed Mahdi Rezaeinia, Ali Ghodsi, and Rouhollah Rahmani. "Improving the Accuracy of Pre-trained Word Embeddings for Sentiment Analysis". In: *CoRR* abs/1711.08609 (2017). arXiv: 1711.08609. URL: http://arxiv.org/abs/1711.08609.

[26] Sean Robertson. *PyTorch Tutorials: Translation with a Sequence to Sequence Network and Attention*. 2017. URL: https://pytorch.org/tutorials/intermediate/seq2seq_translation_tutorial.html (visited on 05/24/2018).

[27] William A. Scott. "Reliability of Content Analysis:The Case of Nominal Scale Coding". In: *Public Opinion Quarterly* 19.3 (1955), pp. 321–325. DOI: 10.1086/266577. eprint: /oup/backfile/content_public/journal/poq/19/3/10.1086/266577/2/19-3-321.pdf. URL: http://dx.doi.org/10.1086/266577.

[28] Gunter Senft. "Phatic communion". In: *Culture and language use*. Ed. by Gunter Senft, Jan-Ola Östman, and Jef Verschueren. Amsterdam/Philadelphia, 2009, pp. 226–233.

[29] Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. “A machine learning approach to coreference resolution of noun phrases”. In: *Computational linguistics* 27.4 (2001), pp. 521–544.

[30] Duyu Tang, Bing Qin, and Ting Liu. “Document Modeling with Gated Recurrent Neural Network for Sentiment Classification”. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 2015, pp. 1422–1432.

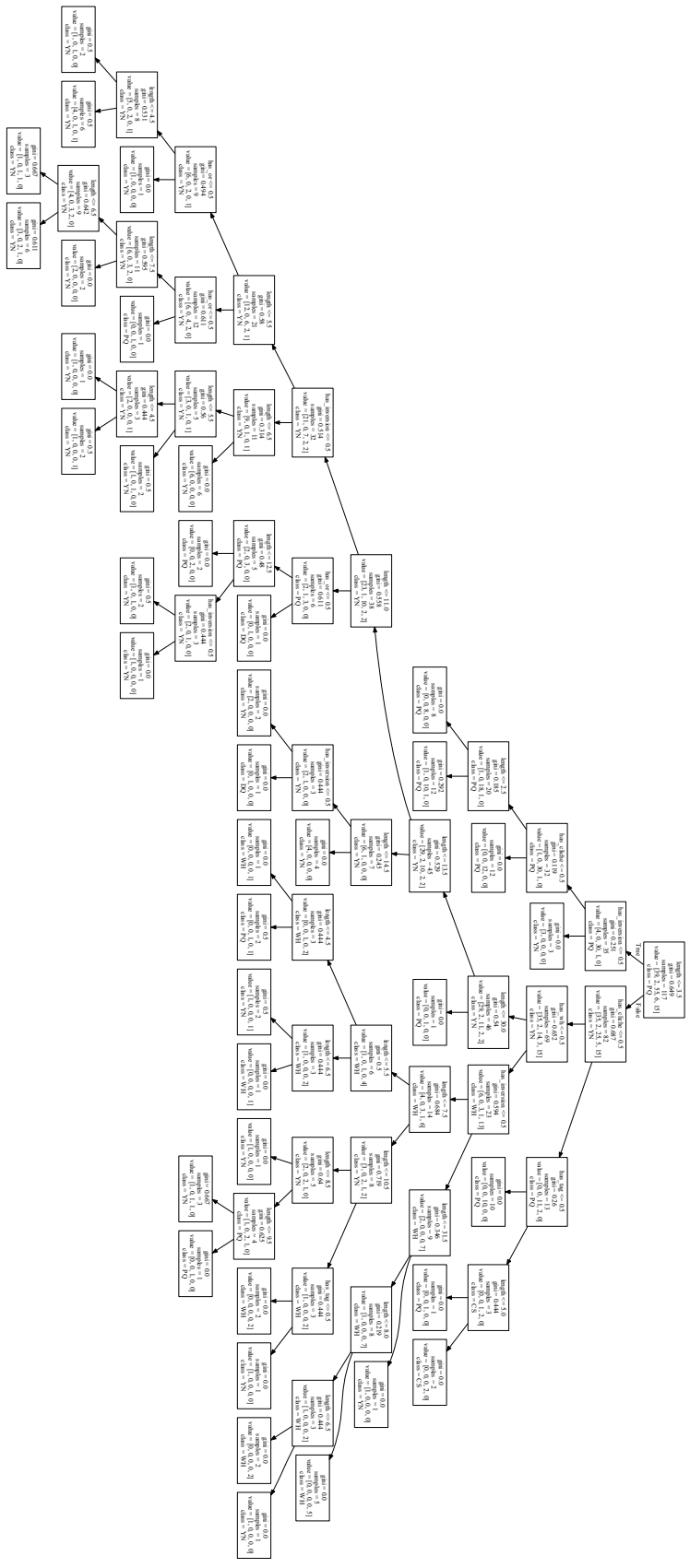
[31] Jiliang Tang, Salem Alelyani, and Huan Liu. “Feature selection for classification: A review”. In: *Data Classification: Algorithms and Applications* (2014), p. 37.

[32] Stéphane Tiv. “Modeling Dialogue in Dynamic Framework”. MA thesis. Université de Lorraine, 2016.

[33] Francisco J. Valverde-Albacete and Carmen Peláez-Moreno. “100Normalized Information Transfer Factor Explains the Accuracy Paradox”. In: *PLOS ONE* 9.1 (Jan. 2014), pp. 1–10. DOI: 10.1371/journal.pone.0084217. URL: https://doi.org/10.1371/journal.pone.0084217.

[34] Dell Zhang and Wee Sun Lee. “Question Classification Using Support Vector Machines”. In: *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*. SIGIR ’03. Toronto, Canada: ACM, 2003, pp. 26–32. ISBN: 1-58113-646-3. DOI: 10.1145/860435.860443. URL: http://doi.acm.org/10.1145/860435.860443.

A. Decision tree output



B. Annotation guide

On the following pages, we include the complete version of our annotation guide.

What's The Answer: Dialogue Annotation

809 | Supervised Project | Annotation Guide

María Andrea Cruz Blandón Gosse Minnema Aria Nourbakhsh

May 27, 2018

Contents

1 Introduction

An important part of corpus annotation is to provide a concise and, at the same time, general guideline appropriate for the task. This guideline explains different procedures that an annotator should take into account for the task of annotation of question and answers in natural language dialogues. The current version of the guide focuses on the English language, and the SCoSE corpus in particular, but for use in future studies it could be easily adapted for different languages and corpora.

2 Setting up your environment

Before going through the description of the annotation process, it is necessary to set up the working environment. This section explains how to install and set up the annotation tools and download the corpus.

2.1 Obtaining corpus files

The SCoSE corpus can be downloaded from https://ca.talkbank.org/access/SCoSE.html.

Although it will be enough to download the transcriptions, it is recommended to also download the corresponding audio files. Listening to the audio files makes the annotation task easier, and, more importantly, there are some cases in which listening to the intonation pattern will help resolve ambiguities in the transcription.

All the transcriptions of the corpora used in this project use the *.cha* format for the dialogues files. It is recommended to read the Symbol Summary document [1] to understand what the symbols in the dialogues represent.

2.2 Setting up ELAN

The annotation tool used is ELAN (ELAN Linguistic Annotator)[2] which allows annotating video and audio files. ELAN can open `.cha` natively, therefore, the files do not need any extra modifications or reformatting.

To install ELAN, please follow the instructions on the official website https://tla.mpi.nl/tools/tla-tools/elan/download/. ELAN is available for Windows, Linux and macOS. Once the tool has been installed correctly, it is time for setting up:

- Step 1: Importing files

Launch ELAN, click on the menu 'File' and then click on 'Import'. Select 'CHAT files'. Choose one of the `.cha` files of the downloaded corpora. See figure 1. After selecting the transcript file, ELAN will allow you to add the media file, though this step is not required.

- Step 2: Create Layers

We define layers to separate the different annotations; these layers represent an abstraction of distinct levels of information, for instance, question types and answer types. In ELAN the representation of layers are tiers. It is possible to define different layers that will build a modular structure; later this structure can be used as a direct extraction method of a particular segment.

This project uses five layers: QUESTION_TYPE, ANSWER_TYPE, FEATURE, COMPLEXITY, and IS_QUOTED:

Question_Type This layer contains the annotations related to question type. Please see section 3.1.1 for more details about the tag set for this layer.

Answer_Type This layer contains the annotations related to answer type. Please see section 4.1.1 for more details about the tag set for this layer.

Feature This layer should be used in case of a question of type *DQ* or *WH*. It is related to the annotation of features. Please see section 3.1.2 for more details about the tag set for this layer.

Complexity This layer is used to identify multiple questions. Please see section 3.1.3 for more details about the tag set for this layer.

Is_Quoted This layer will contain the annotations related to direct and indirect speech. It is used for both questions and answers. Please see section 3.1.4 for more details of possible values.

[1] http://www.bu.edu/linguistics/UG/course/lx865-f02/local/childes-symbols.pdf
[2] https://tla.mpi.nl/tools/tla-tools/elan/

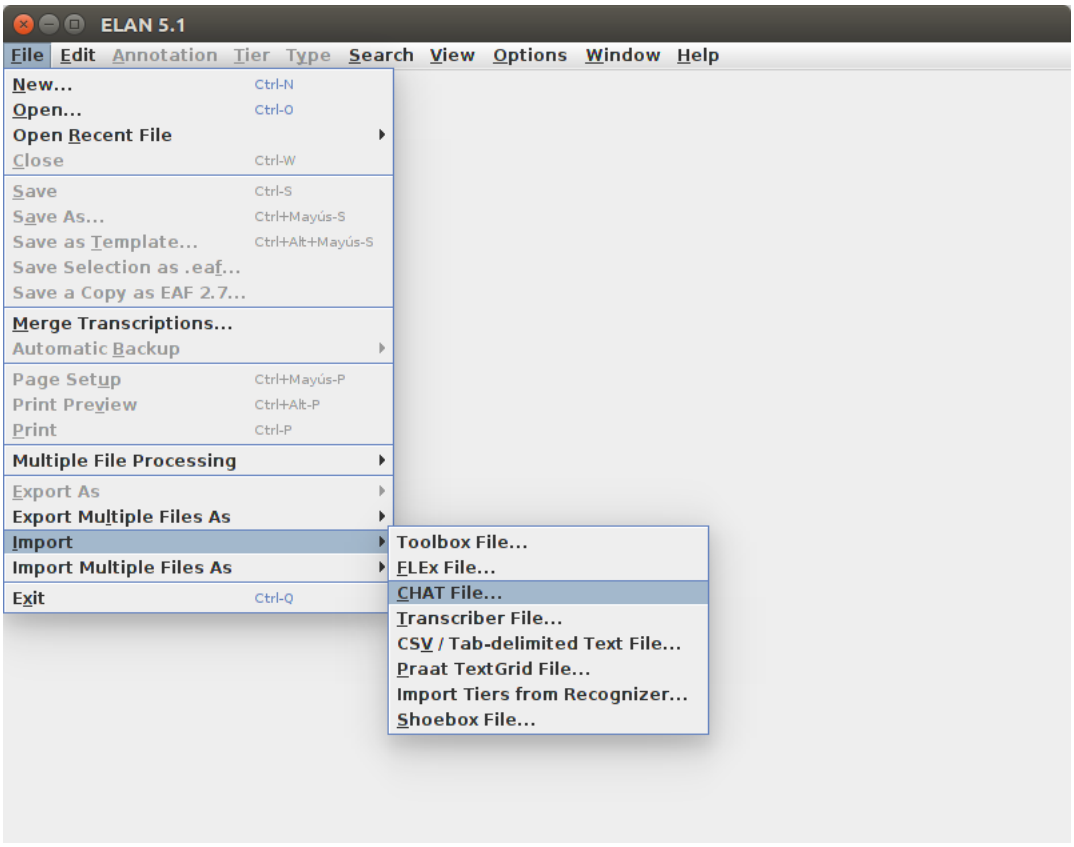


Figure 1: How to import a file from the corpora

The following steps is for creating all these layers in ELAN. To create a layer click on the 'Tier' menu, and then on 'Add New Tier'. In the opened window ('Add Tier'), fill out the form as follows (see figure 2):

- *Tier Name*: enter the name of the layer;
- *Annotator*: enter your name;
- *Input Method*: select 'English (ASCII)'.

Leave the default values in the rest of the fields. Now click on the 'Add' button to create the layer. It is possible to create all of the layers without closing the window.

- Step 3: Annotate

Before continuing, save the document: click on the menu 'File', and then on 'Save'. This step will create an EAF file (`.eaf`), which is ELAN's native file format. ELAN allows to make partial annotations and resume it later using the ELAN file. Before closing the file, always save your work so your annotations will be available for the next time.

To annotate, you should first select the timespan which corresponds to the utterance you want to annotate. Next, you should choose the appropriate tier by clicking on it. Once you have clicked on the layer you now enter the tag, as is shown in figure 3.

3 Tagging questions

This section describes how you should tag questions. First, in section 3.1, we explain our tag set, and then, in section 3.2, we provide a step-by-step description of the tagging process.

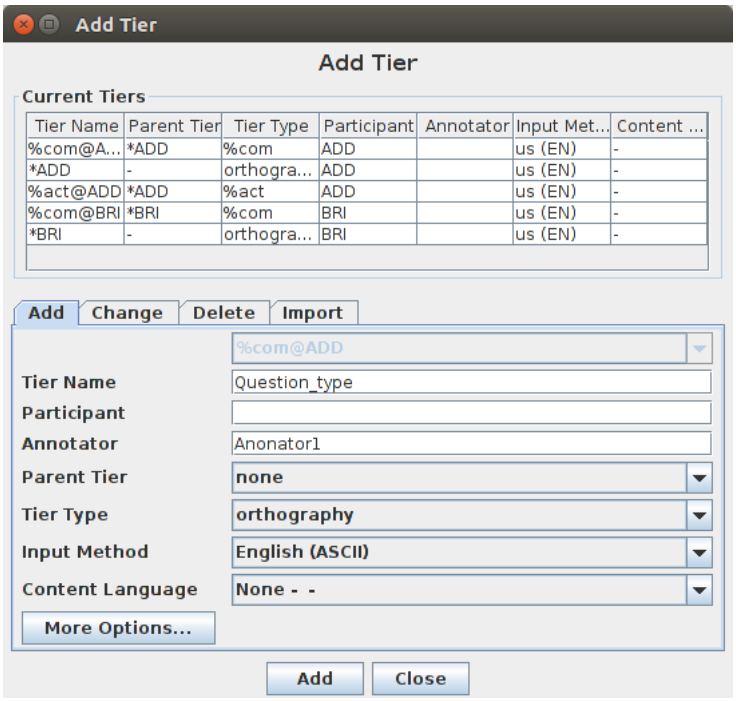


Figure 2: How to create a layer

3.1 Tags

Our annotation scheme uses several layers, each of which is associated with its own set of tags. In this section, we define these tag sets, layer by layer.

3.1.1 Question types

Summary: Classification of questions according to formal and functional criteria. Layer: QUESTION_TYPE. Tags: *YN* (yes/no question), *DQ* (disjunctive question), *PQ* (phatic question), *CS* (completion suggestion), *WH* (wh-question)

In this section, different types of questions are explained, a formal definition is provided and for each one of them some examples are included. We see each question type as a combination of a specific **form**, i.e., a set of syntactic characteristics, and a specific **function**, i.e., the role that a question plays in the discourse.

1. **Yes/no question *YN***): In these types of question, a proposition is expressed and the expected answer will confirm or deny this proposition. Yes/no questions look for a real answer from the other participant of the conversation. Questions of this type have the form 'yes/no' the function 'non-phatic' to distinguish it from phatic questions (type #3). Hence, in our schema, just having the form of a yes/no question is not sufficient to be classified as a 'real' yes/no question.

   Definition 1. A yes/no question asks the other participant to confirm or deny a proposition.

   #### Prototypical examples

   (1) Or will you be just gonna hanging around with Brianne? (SCoSE/Amy, 35)

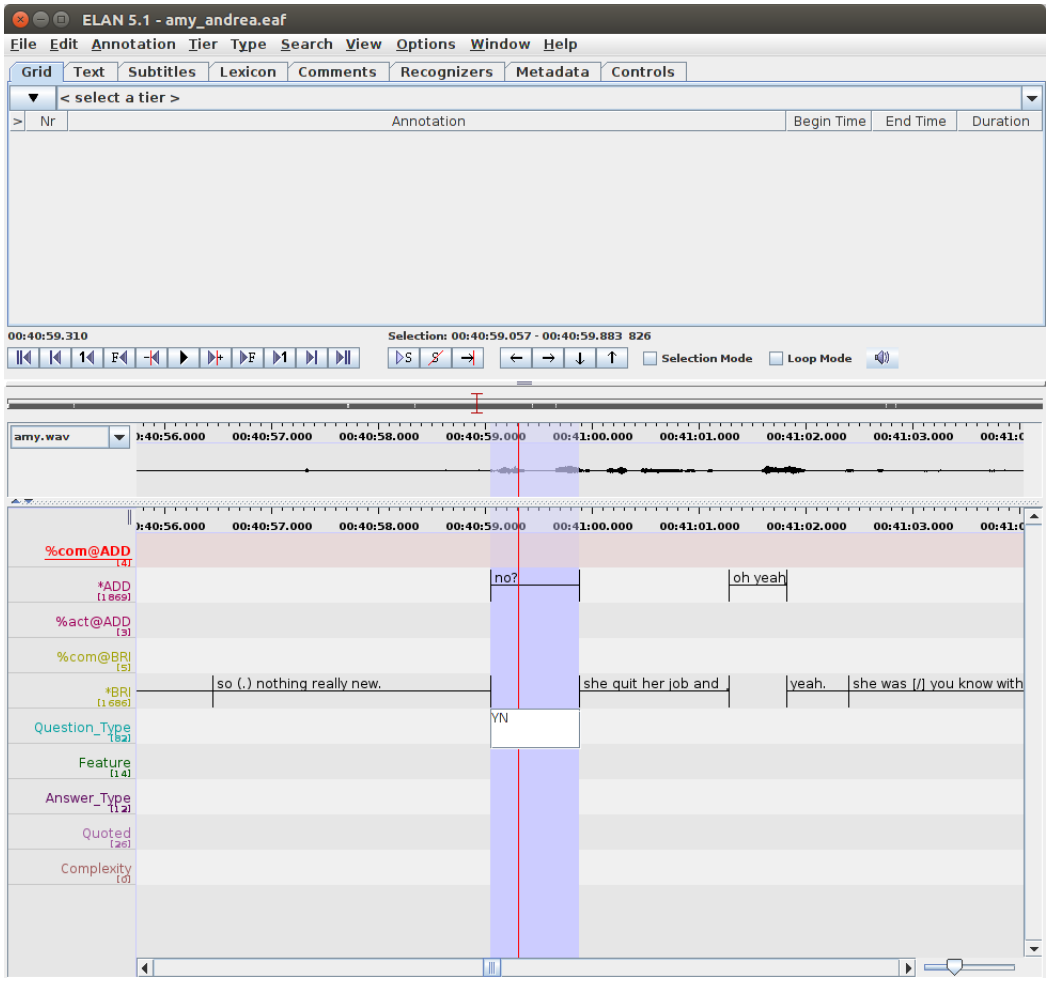


Figure 3: How to annotate an utterance

(2) Oh d'you have a white shirt? (SCoSE/Amy, 130)

(3) Does it? (SCoSE/Amy, 70)

2. **Disjunctive question (*DQ*)**: Questions of this type usually contain "or" to distinguish between different options which are proposed to the hearer. The hearer can choose one of the options or state a third option of his own. The function of these questions is to ask for a feature, which should be annotated on its own tier (see 3.2).

   Definition 2. A disjunctive question asks the hearer to choose between one of several options.

   ### Prototypical examples

   (4) Is the wedding on Sunday or is it Saturday? (SCoSE/Amy, 227)

   (5) Do you want coffee or tea?

   Note. Disjunctive questions always expect a feature as an answer. This means that some questions containing 'or' (e.g. questions ending in 'or not?') are not disjunctive questions.

3. **Phatic question (*PQ*)**: Phatic questions are asked to continue participation in communication or to indicate that the listener follows the conversation or to express surprise,

rather than to ask for some kind of information. Phatic questions often contain clichés like 'you know?' or 'right?', but in principle they can have any form and are distinguished by their function rather than by their form.

Definition 3. Phatic questions are questions that have a purely communicative purpose and that do not expect an informative answer.

Prototypical examples

(6) Oh really? (SCoSE/Amy, 28)

(7) Right? (SCoSE/Amy, 120)

Note. In some cases, it is hard to decide if a question expects an informative answer or not, and hence, whether it is a phatic question or a question of one of the other types (e.g. YN, WH). Cues from prosody and the discourse context can be helpful for making this distinction.

4. **Completion suggestion (*CS*)**: Sometimes, in the middle or at the end of an utterance, the other participant of a conversation asks a question in a way to suggest an option or multiple options to the first speaker so he/she can complete his/her statement with the information provided in the question. One typical situation is when the speaker hesitates to say what he/she means and the one who asks the question tries to help him/her to finish his utterance.

   The form of a completion suggestion can vary, but in all cases, it logically continues the previous utterance, and often, it interrupts this utterance.

   Definition 4. Completion suggestions are questions that propose an expression to the other participant to be used to finish his/her statement.

Prototypical examples

(8) A: It includes heat and, uhm, I think...
B: Water? (SCoSE/Amy, 747)

5. **Wh-question (*WH*)**: This type of question is easy to spot as it always contains a wh-constituent. Usually, this constituent, introduced by a wh-word (like what, who, where, when, which, how,...), appears at the beginning of the question. Wh-questions ask for a feature which should be annotated for its function.3.1.2.

   Definition 5. A wh-question is a question that contains a wh-constituent (introduced by a wh-word like what, who, where, when, which, how).

Prototypical examples

(9) What are your plans now? (SCoSE/Amy, 241)

(10) When will you guys get off? (SCoSE/Amy, 243)

3.1.2 Features

Summary: Only for *WH*, *DQ*; indicates the semantic function wh-phrase/disjuncts/expected answer. Layer: FEATURE. Tags: • *TMP* (temporality) • *LOC* (location) • *AG* (agent) • *TH* (theme) • *OW* (owner) • *RE* (reason) • *CH* (characteristic)

Both the wh-phrase in a wh-question, and the disjuncts in a disjunctive question, have a particular semantic role in the question; the expected answer should contain a constituent with the same role. For example, in '*When* did you leave the party?', the wh-phrase 'when' denotes the time when the addressee left the party, and an appropriate answer to this question (e.g. 'around midnight') should also have this semantics. A similar example for a disjunctive question would be 'Did you leave the party before midnight or later?'.

In our annotation scheme, semantic roles are called 'features'. All wh-questions and disjunctive questions (but no questions of any of the other types) should be annotated with a tag, on the FEATURE layer, indicating their respective feature. Answers should never be tagged with a feature; this would be redundant since the feature applies both to a constituent in the question and to the corresponding constituent in the answer. There are eight different tags for features:

1. Temporality (*TMP*): the constituent refers to a moment or a period related to the event described in the question;
2. Location (*LOC*): the constituent refers to a location related to the event described in the question;
3. Agent (*AG*): the constituent refers to the person (or, less commonly, the entity) that performed the action described in the question;
4. Theme (*TH*): the constituent refers to the person or entity that underwent the action described in the question;
5. Owner (*OW*): the constituent refers to the person (or other entity capable of ownership, such as an organization) who owns (in a broad sense that also includes, for example, family relations) the entity that is described in it;
6. Reason (*RE*): the constituent refers to the the reason or motive behind the event described in the question;
7. Characteristic (*CH*): the constituent refers to a characteristic of the event described in the question.

In wh-questions, the feature of the wh-constituent is closely linked to the wh-word that is used. These correspondences are summarized in Table 1.

3.1.3 Complexity

Summary: Indicates if the utterance contains one or more questions. Layer: COMPLEXITY. Tags: • *SQ* (single question) • *MQ* (multiple questions)

In some cases, a single utterance, or even a single sentence, can comprise multiple questions. These questions do not necessarily have the same type and should therefore be tagged separately; however, since they are part of the same utterance they have a strong pragmatic relationship, they should also be tagged as a whole. Hence, we define two types of utterances:

1. Single-question utterances (*SQ*): utterances that contain only one question;
2. Multi-question utterances (*MQ*): utterances that contain at least two questions.

Wh-word	Features	Code
What + focus phrase	Feature(focus phrase)	
When	TEMPORALITY	TMP
Where	LOCATION	LOC
Who	AGENT	AG
Whom	THEME	TH
Which + focus phrase	Feature(focus phrase)	
Whose	Owner	OW
Why	Reason	RE
How	Characteristic (Ch.)	CH

Table 1: Wh-words and features

Obviously, there also exist utterances which contain no question at all, but these should not be tagged.
Note. We define an 'utterance' as a speech fragment uttered by a single speaker. In the .cha version of the transcriptions, the beginning of each utterance is marked by `*SPEAKER:` (where `SPEAKER` is the speaker's code); in ELAN, each utterance is marked as a single annotation. An utterance consists of one or more sentences (each of which ends in a period, a question mark, or an exclamation mark).

3.1.4 Quotations

Summary: Indicates if the question is used in direct or in indirect (quoted) speech. Layer: IS_QUOTED. Tags: • *QQ* (quoted question) • *NQ* (non-quoted question)

Conversations sometimes include quotations; for example, a speaker might be recounting a conversation that they had earlier with someone else, or a conversation between other people that they overheard. We call this 'quoted speech': it is speech that is reporting other people's speech, rather than speech that is directly aimed at the other discourse participants. 'Non-quoted speech' is all other speech: it is speech that is not part of a quotation. Questions can occur in both types of speech, and should be classified accordingly:

- Quoted questions (*QQ*): questions that are part of a quotation, and, as such, are not expected to be answered by the addressee.
- Non-quoted questions (*NQ*): all other questions

Note. Although *QQ* questions are not expected to be answered by the addressee, this does not necessarily mean that there will be no answer at all: in some cases the speaker might also recount the answer that was given in the original discourse (e.g. 'They asked me …, and then I said …').

3.2 Tagging procedure

When annotating (part of) a corpus, go through it line by line. Each line should be treated as a separate utterance. For each utterance, follow the following steps:

- Step 1: Identifying questions

Summary: Only annotate utterances that end in a question mark (in the transcription).

The first step is to identify whether or not the utterance contains at least one question. Note that, in principle, this is not a trivial task: there are many factors (including syntax, prosody and the pragmatic context) that determine whether or not the utterance has an interrogative meaning. It would be too complex and time-consuming to do a full analysis of all of these factors for each sentence, so you should not attempt to do this. Instead, simply use the analysis of the original transcribers of the corpora: all utterances that they considered to be questions are marked with a question mark at the end of the sentence.[3]

- **If you find a question mark:** go to step 2. 5
- **Otherwise:** skip this utterance and go to the next one.

Note (1). Beware that transcriptions can and do contain mistakes; in some cases, questions marks might have been placed or left out in error. If an utterance is not ended by a question mark, never tag it, even if you believe that the question mark was omitted erroneously. If there is a question mark, and if you think that this might be a transcription error, first analyze the utterance carefully with the help of the media files of the corpus (if available) and try to rule out that this utterance is a question with a non-canonical form (e.g. a yes/no question without inversion) or a non-canonical function (e.g. a rhetorical question). If you still believe that it is a transcription error, ignore the utterance and mark in the log that you did so.
Note (2). Even if only part of an utterance is interrogative, the entire utterance should be tagged as a question.

Example 1. The following sentences are ended by question marks, and from their semantics it is obvious that they are indeed questions:

(11) Oh really? (SCoSE/Amy, 28)

(12) What colour is it? (SCoSE/Amy, 75)

Example 2. Question (13) lacks a question mark, and as such should not be tagged, even its semantics would justify this.

(13) *Cómo se llama.* 'What is his name.' (TalkFriend/Spanish, 274)

Example 3. (14) is an example of an utterance that, judging by the transcription, clearly looks like a question. However, when taking into account the intonation of this utterance (using the audio recordings), it becomes clear that the transcription is mistaken: in fact, the utterance that was produced is not a question, but an affirmative sentence (*'Si ya no estarían trabajando los dos sino uno.'*) followed by a discourse marker that expresses regret on the part of the speaker (*'Noo...'*). Confusingly, in Spanish, *'no'* can also be used at the end of (tag) questions, but this *'no'* has a very different intonation than *'no'* as a discourse marker, and it is easy for a native speaker to detect the difference.

(14) *Si ya no estarían trabajando los dos sino uno, no?* 'Yes, they would no longer be working together but only one, no?' (TalkFriend/Spanish, 642).

[3]Note that this notation is sometimes different from the normal convention for marking questions in written language. For example, in written Spanish, questions not only have a question mark at the end, but also an inverted question mark at the beginning. In the TalkBank corpora this is not the case.

- Step 2: Determining complexity

Summary: Check if the utterance contains multiple questions (if so, annotate *MQ*) or just one question (if so, annotate *SQ*).

Look for signs that the utterance contains more than one question. If the utterance comprises more than one sentence, and if there is more than one question mark, this is a strong indication that there are multiple questions. Also, if you see a question that contains a coordination (typically with 'and'), check if each of the coordinates is a question of its own.

- **If there is more than one question:**
  1. Mark the entire sentence with *MQ* on the COMPLEXITY layer.
  2. For each sub-question, find and mark its timespan.
  3. For each sub-question, execute step 3 and subsequent steps.
- **Otherwise:** go to step 3.

Example 4. The utterance in (15) contains two questions, 'And him in what?' and 'In golf?'. The complete utterance should be tagged with *MQ* on the COMPLEXITY layer. The first part should be tagged as *WH* (QUESTION_TYPE), *TH* (FEATURE), and *DS* (COMPLEXITY). The second part should be tagged as *YN* (QUESTION_TYPE) and *DS* (COMPLEXITY).

(15) *Y e-, él, él en qué? En golf.*[4] 'And him in what? In golf?' (TalkFriend/Spanish, 557)

- Step 3: Assigning question types

Summary: Check the definitions of *WH, DQ, YN, CS, PQ* and apply the first matching tag.

Go through all of the question type definitions and check if the current question matches that definition. Once you have a match, stop and tag the question accordingly. If nothing matches, assign the last tag on the list (i.e., *PQ*), unless you have a very strong suspicion that the utterance you are looking at is not in fact a question (i.e. that the question mark in the transcription was added erroneously) – see the note in step 1.

Use the following order for trying definitions:

1. Wh-questions (*WH*)
2. Disjunctive question (*DQ*)
3. Yes/No questions (*YN*)
4. Completion suggestion (*CS*)
5. Phatic questions (*PQ*)

- **If you assigned *DQ* or *WH*:** go to step 4.
- **Otherwise:** go to step 5.

For examples for each of the question types, go to section 3.1.1.

- Step 4: Assigning feature types

Summary: For wh-questions with *when, where, who, whom, whose, why, how*: choose the feature from the list. In all other cases, find the most likely feature based on semantics.

[4] Note that the second question (very likely mistakenly) does not end in a question mark, so it should not be tagged; the tags given here are for illustration purposes only and apply to the hypothetical situation in which it would have been marked with a question mark.

- **For wh-questions (*WH*):** first find the main wh-phrase in the sentence. The 'main' wh-phrase is defined as:
  (a) The fronted wh-phrase (in English and other languages with wh-movement), if there is one;
  (b) More generally, the wh-phrase that, according to your semantic judgement, refers to the piece of information that the asker of the question would like to know.
  - If the main wh-phrase starts with *when, where, who, whom, whose, why,* or *how* (in English) or the translation equivalent of one of these wh-words in a different language, choose the corresponding feature from Table 1 (page 8).
  - If the main wh-phrase starts with *what* or *which* plus a 'focus phrase' (e.g. 'what *[book]*', 'which *[wealthy country]*'), choose the feature from the table that comes closest to describing the semantic role of the focus phrase. Do this by using your own semantic judgment, but look, where possible, for clues in the syntactic structure of the sentence (see the note at the bottom of this step).
  - If the main wh-phrase only consists of *what*, without a focus phrase, choose the feature from the table that comes closest to describing the semantic role of the wh-phrase. Again, use your semantic judgement and syntactic cues.
- **For disjunctive questions (*DQ*)**: identify the disjuncts in the sentence. Then, choose the feature from the table that comes closest to describing the semantic role of these disjuncts. Again, use your semantic judgment and clues from the syntax of the wh-phrase (see the note below).

Go to step 5.

Note (1). If the semantic role of the phrase that you are tagging does not correspond to any of the feature type definitions, default to the tag *TH*.

Note (2). The following syntactic clues can be helpful for identifying the feature of a focus phrase or of a disjunct in a disjunctive question:

(a) *Syntactic role/position:* agents (*AG*) are often subjects; themes (*TH*) are often objects.

(b) *Prepositions:* in prepositional wh-phrases, the preposition (or postposition) can sometimes give clues about the semantic role of the phrase (but note that some are ambiguous):
  - 'in', 'at', 'under', 'behind', … ⇒ location (*LOC*)
  - 'in', 'at', 'after', 'during', … ⇒ temporality (*TMP*)
  - 'because of', 'thanks to', … ⇒ reason (*CH*)

Example 5. In (16), the wh-constituent is easy to find: it is 'what', located at the front of the sentence. It is not followed by a focus phrase, so we have to try to find the semantic role of *what* itself. In this case, *what* represents the contents of the plans of the addressee of the question; the semantic role that comes closest to describing this is theme (*TH*).

(16) What are your plans now? (SCoSE/Amy, 241)

In (17), the wh-constituent, *where*, is not fronted, but is still easily identifiable. For *where*, the feature can be found in the table: it is 'location' *LOC*.

(17) To where? (SCoSE/Amy, 2483)

- Step 5: Determining 'is quoted' status

Summary: If the transcription indicates quoted speech (or if there are other strong clues): annotate *QQ*; otherwise annotate *NQ*.

Check if the question you are looking at was uttered in the context of direct or of indirect speech. To do this, look for clues in the transcription. Indirect speech is indicated in one of the following ways:

(a) It starts with `` and ends with '';

(b) It is introduced by +";

(c) Is is ended by +".

(d) The preceding utterance is ended by +"/.

If the transcriptions indicate that an utterance, or part of an utterance, is quoted, always annotate it as such. However, if there is no such indication, it is still possible that the utterance is quoted speech. If you believe that this might be the case, use the following two factors to guide your decision:

(a) *Prosody:* listen carefully to the audio recording of the utterance (if available). If the speaker is using an unusual intonation (e.g. imitating someone else's voice), this could be an indication of indirect speech.

(b) *Pragmatic context:* if the question you are looking at is part of a story (or conversation) that the speaker is telling from the perspective of someone else, it is likely that it is indirect speech. However, be careful: it is also possible that the speaker interrupts their story to ask a question 'as themselves'; in this case, the question is obviously not indirect speech.

- **If the question is (part of) indirect speech:** annotate *QQ* on the QUOTED layer.
- **Otherwise:** annotate *NQ* on the QUOTED layer.

Example 6. (18) is quoted speech: it is marked with quotation marks and from the context it is clear that the speaker is not asking this question to her conversational partner, but is simply reporting that someone else asked this question. By contrast, (19) is not marked with quotation marks and is asked directly to the speaker's interlocutor.

(18) "So you're gonna be home then?" (SCoSE/Amy, 33)

(19) What colour is it? (SCoSE/Amy, 75)

- Step 6: Checking your annotations

Check your own annotations for the current question. Please pay attention to the following:

- Check that you included a tag in each layer that you needed to tag (e.g., if you annotated a question as WH or DQ, make sure that you also include a feature);
- Check that every tag was placed in the correct layer;
- Check that there are no typos in the tags that you used

Question_Type (Code)	Anwer_Type (Code)	Description
YN, CS	PA	Positive Answer
YN,CS	NA	Negative Answer
DQ, WH	FA	Feature Answer
YN, CS, DQ, WH, PQ	PHA	Phatic Answer
YN, CS, DQ, WH, PQ	UA	Uncertain Answer
YN, CS, DQ, WH, PQ	UT	Unrelated Topic
YN, CS, DQ, WH, PQ	DA	Deny the Assumption

Table 2: Possible answers types according to question types

4 Tagging answers

This section explains how to annotate answers. First in section 4.1.1 we explain the different tags and in section 4.2 we explain the process you should follow to tag the answers.

4.1 Tags

This section contains the description of each type of answers and the tags that correspond to those types.

4.1.1 Answer types

Summary: Classification of answers. Layer: ANSWER_TYPE. Tags: *PA* (positive answer), *NA* (negative answer), *FA* (feature answer), *PHA* (phatic answer), *UA* (uncertain answer), *UT* (unrelated topic), and *DA* (deny the assumption)

In this section, different types of answers are explained, a formal definition is provided and for each one of them some examples are included. Contrary to the questions, we do not make a distinction between form and function. Instead, answer types are based on the types of questions that they respond to. The correspondences between question types and answer types are summarized in table 4.1.1.

1. **Positive answer (*PA*)**: This type of answer is expressed to confirm the proposition contained in the question. Usually, but not always, positive answers include positive tokens (such as yes, right, okay, sure ah, oh yeah,...). Positive answers are expressed in response to yes/no and completion suggestion questions.

   Definition 6. A positive answer is an utterance that indicates that the proposition of a question is true.

   #### Prototypical examples

   (20) Q: Are you still friends with all those same people if it doesn't work out?
   A:Exactly (SCoSE/Amy, 2707- 2808)

   (21) Q: Yeah, do you really?
   A: Yeah! (SCoSE/Amy, 218-219)

   Note. If an answer expresses some uncertainty, but is clearly more positive than negative, it should still be tagged as a positive answer.

2. **Negative answer (*NA*)**: Negative answers are opposite of positive answers. This type of answer denies the proposition contained in the question. It is very likely that the answer includes negative tokens (such as no, not, never,...). Negative answers are expressed in response to yes/no and confirmation suggestion questions.

   Definition 7. A negative answer is an utterance that indicates that the proposition of a question is false.

   Prototypical examples

   (22) Q: Are you going too?
   A: Oh I can't. (SCoSE/Amy, 323- 324)

   (23) Q:do you know anyone that goes there?
   A: No! (SCoSE/Amy, 218-219)

   Note. If an answer expresses some uncertainty, but is clearly more negative than positive, it should still be tagged as a negative answer.

3. **Feature answer (*FA*)**: In feature answers, the speaker refers to the feature that has been asked for in the question. This type of answer answers either a wh-question or a disjunctive question.

   Definition 8. A feature answer expresses the feature or features that have been asked in the question.

   Prototypical examples

   (24) Q: What colour is it?
   A: It is midnight blue. (SCoSE/Amy, 75- 76)

4. **Phatic answer (*PHA*)**: Phatic answers play a role that is similar to that of phatic questions. Phatic answers can be expressed in response to any type of question.

   Definition 9. Phatic answers are utterances with a purely communicative purpose which do not have any informational content.

   Prototypical examples

   (25) Q: So what is the point?
   A: Yeah! (SCoSE/Amy, 3497- 3498)

5. **Uncertainty answer (*UA*)**: These type of answers is used when the speaker does not know the answer or cannot provide the information that is asked in the question. This type of question often contains expressions like 'I don't know' or 'I am not sure'.

   Definition 10. An uncertainty answer indicates that the speaker does not know or cannot provide the information that is asked in the question.

Prototypical examples

(26) Q: Is the wedding on Sunday or is it Saturday?
A: It's ... I don't even know. (SCoSE/Amy, 227- 228)

Note (1). In some cases, an answer expresses uncertainty, but also provides the information that the question asked for. In such cases, the answer is not an uncertainty answer but has the type of the provided information.

Note (2). A disjunctive answer to a yes/no question should be considered to be an uncertainty answer. For example, in (27), the first part of the answer ('that') confirms the proposition in the question, while the rest of the answer ('or else ...') denies it and proposes an alternative. Overall, the answer does not confirm or deny the proposition and should be tagged as an uncertainty answer.

(27) Q: You go through four years of regular school, and then to art school or what?
A: That, or else I transfer to a university that has a stronger art department. (SCoSE/Amy, 908-909)

6. **Unrelated topic (*UT*)**: Unrelated topic answers are those answers whose informational content is not related to the question in a direct way. In other words, the question asks some particular information and the answer does not provide that required information directly or the answer speaks about a completely different proposition.

   Definition 11. Unrelated topic answers are answers that are not related to the question.

   Prototypical examples

   (28) Q: when will you guys get off?
   A: my last exam is like. I don't know.[5] (SCoSE/Amy, 243- 244)

7. **Deny the assumption (*DA*)**: 'Deny the assumption' answers are expressed when the question has some presupposition about a phenomenon and the answer negates that presupposition. For example, the question 'have you stopped smoking?' presupposes that you used to be a smoker, even if it is answered negatively. By answering along the lines of 'I never smoked', you denied this presupposition. 'Deny the assumption' answers are quite different from negative answers, although, in many cases, they have a similar form (e.g. they contain a negation); which of the two types applies in a specific situation depends on the context.

   Definition 12. Deny of assumption answers, deny the assumption that the question has taken into account.

   Prototypical examples

   (29) Q: what are your plans now?
   A: I don't have any plans. (SCoSE/Amy, 241- 242)

[5] In this particular example, 'I don't know' is an expression about the date of the exam and not an answer to the question

4.1.2 Quotations

The definitions and tags as described in section 3.1.4 apply to questions as well as to answers.

4.2 Tagging procedure

- Step 1: Identifying answers

Summary: The answer should be a single utterance that occurs shortly after the corresponding question, is pragmatically and/or semantically related to the question, and is not itself a question.

The hardest part of answer tagging is identifying what and where exactly the answer is. What you should do is try to find a single, complete utterance that satisfies the following conditions:

- The utterance should be uttered shortly after the question.
  - Unfortunately, 'shortly' cannot be defined objectively. In many cases, the answer will be the utterance directly after the question, but in some cases, it can be one or a few utterances after that.
  - In some cases, the answer will interrupt the question, i.e., the answer will be given while the question is still being formulated.
- The utterance should be pragmatically and/or semantically related to the question that it answers.
  - Prototypically, the answer will be uttered in response to the question (pragmatics) and contain the information that was requested in the question (semantics).
  - In some cases, the utterance can be indirect, or be semantically unrelated to the question. However, if it is uttered as a response to the question, it should still be considered to be an answer.
  - If the utterance is uttered in response to either an external stimulus or to an utterance other than the question, it should not be tagged as an answer.
- The utterance is not itself a question.
  - If a question is directly followed by another question, this second question should always be tagged as a question rather than as an answer, even if it is asked in response to the earlier question.
- Interjections (e.g. 'uhm', 'mhm mhm') can also be answers.

Note (1). In some cases, there is no answer at all. In such cases, you do not have to tag anything.
Note (2). The only case in which you should annotate more than one utterance as the answer is in cases where a speaker is interrupted by another speaker, and continues their answer in the next utterance.

Example 7. (30) is an example of a prototypical case where the answer is simply the first utterance after the question. On the other hand, in (31), the situation is more complex: the answer to the question in the first line is given in the third line ('Is it a guy or a girl?'/'It's a guy'), and the answer to the second question is given in the fourth line ('A woman teacher?'/'A guy teacher')

(30) Q: What colour is it?
A: It's midnight blue. (SCoSE/Amy, 75-76)

(31) Q: Is it a guy or a girl?
Q: A woman teacher?
A: It's a guy.
A: A guy teacher. (SCoSE/Amy, 2296-2300)

Example 8. The answer in (28) above is an example of an utterance that does not directly answer the question that was asked, but is clearly uttered in response to it. Utterances like this should be tagged as an answer and assigned the type *UT*. By contrast, in (32), from the context it is clear the utterance that follows the question is uttered in response to a stimulus that is external to the conversation (i.e. the 'handsome man' that the speaker sees), not to the question that was asked; and the question itself is never answered. In such cases, nothing will be tagged as an answer.

(32) Q: *Wat doet-ie?* ('What does he do?')
A: *Ik zie een mooie man.* ('I see a handsome man.') (CGN/fn008003, 91-92)

- Step 2: Assigning answer types

Summary: For answers to *YN, CS* questions, first try *PA, NA*; for *WH, DQ*, first try *FA*; after this, and for other question types, try *DA, UA, UT, PHA* (in that order).

For assigning answer types, first check the type of the question that the answer responds to.

- **If the question has type *YN* or *CS*:** first check if the answer matches the definition of *PA* (Positive Answer) or *NA* (Negative Answer). If either of these matches, apply one of these two tags, as appropriate. If not, go to 'All question types' below.
- **If the question has type *WH* or *DQ*:** first check if the answer matches the definition of *FA* (Feature Answer). If it does, apply this tag. If not, go to 'All question types' below.
- **All question types:** check the following answer types in the specified order, and apply the first one whose definition matches:
  - *DA* (Deny the Assumption)
  - *UA* (Uncertainty Answer)
  - *UT* (Unrelated Topic)
  - *PHA* (Phatic Answer)

For examples of each of the answer types, go to section 4.1.1.

- Step 3: Determining 'is quoted' status

Follow the same steps as in step 5 of the tagging procedure for questions.

- Step 4: Checking your annotations

Follow the same steps as described in step 6 of the tagging procedure for questions.

5 Troubleshooting

This section lists the possible problems you may face while annotating.

- **I have mistakenly created an annotation in a layer. How can I erase the annotation?** Select the annotation, right-click on it and then choose 'Delete annotation'.

- **Two utterances need to be tagged, but in the transcription, they appeared overlapped. How can I annotate both of them?**
  First, identify the shortest one. If this utterance is in the beginning or middle of the overlapping, you should tag from the beginning of the overlap until the end of the utterance, and then begin the second annotation for the longest one. In the other case, you should tag the longest one from the beginning until the second utterance starts and then tag the second utterance. See figure 4.

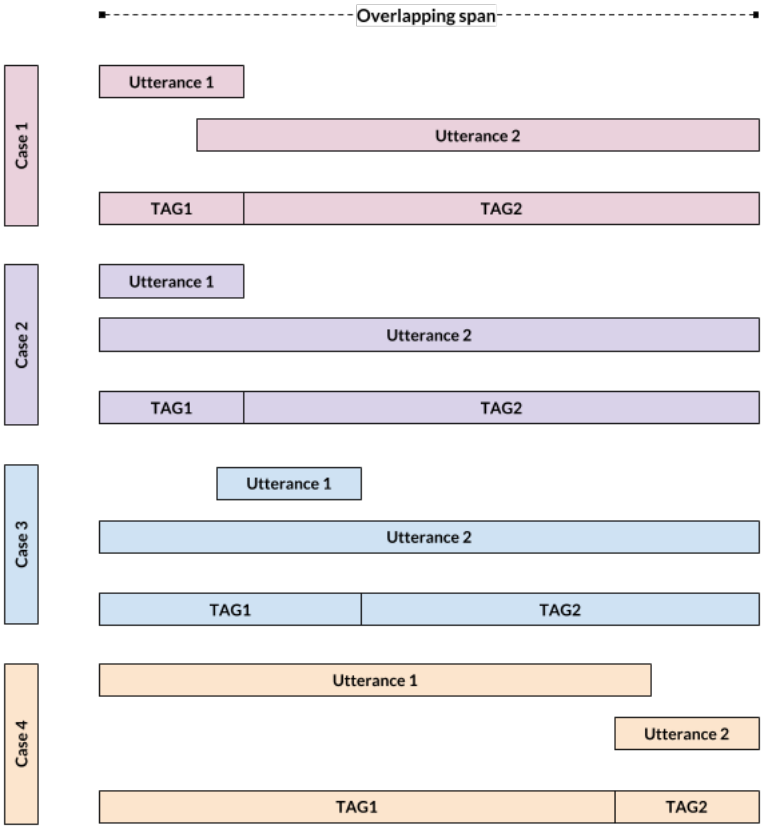


Figure 4: How to annotate overlapping utterances

- **The answer to a question is not the following utterance. (This can be the case when the speaker adds more information or there exists more than one question, and the hearer answers first one of the questions.) Is there any identifier of which answer belongs to which question?**
  There is not such identifier. You should annotate the question and the answer as usual, without extra information.

- **There is an interruption in the question or answer I am annotating. What should I do?**
  The strategy to obtain the whole question/answer will be to enumerate the tags. That is, in the QUESTION_TYPE layer, whenever there is an interruption, you should annotate the question as usual, but add an underscore and number (which will identify the order) starting from 1. For example, you classify the question as a *WH* question, but the question is split into two utterances, then your tags for each utterance that made the complete question will be *WH_1* for the first utterance and *WH_2* for the second one. The Figure 5 illustrates this.

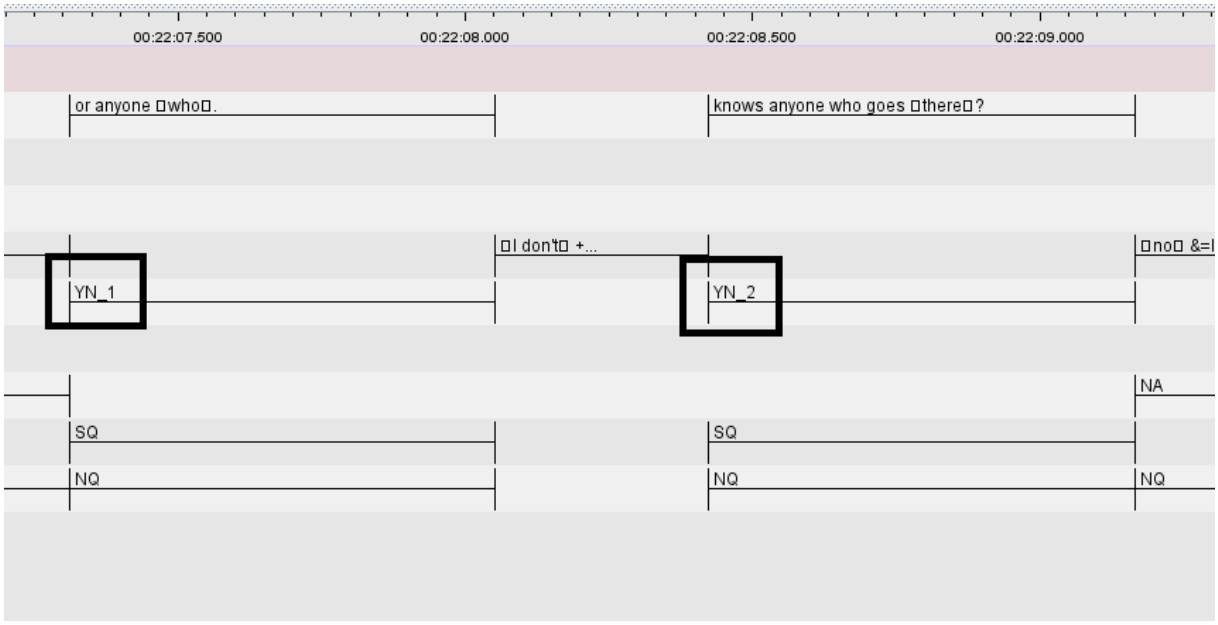


Figure 5: How to annotate interrupted utterances. Question: *or anyone who ...* + *... knows anyone who goes there?*