Zurich University of Applied Sciences

# Automatic Action Item Detection

*Author:*
Flurin Gishamer

*Supervisor:*
Prof. Dr. Mark Cieliebak

Institute of Applied Information Technology
Switzerland

July 31, 2021

Zurich University
of Applied Sciences

School of
Engineering

# DECLARATION OF ORIGINALITY

## Master's Thesis at the School of Engineering

By submitting this Master's thesis, the undersigned student confirms that this thesis is his/her own work and was written without the help of a third party.

The student declares that all sources in the text (including Internet pages) and appendices have been correctly disclosed. This means that there has been no plagiarism, i.e. no sections of the Master's thesis have been partially or wholly taken from other texts and represented as the student's own work or included without being correctly referenced.

Any misconduct will be dealt with according to paragraphs 39 and 40 of the General Academic Regulations for Bachelor's and Master's Degree courses at the Zurich University of Applied Sciences (Rahmenprüfungsordnung ZHAW (RPO)) and subject to the provisions for disciplinary action stipulated in the University regulations.

City, Date: Zürich, 31.7.2021

Signature:

The original signed and dated document (no copies) must be included after the title sheet in all ZHAW versions of the Master's thesis submitted.

## Abstract

Action items are decisions that a group agrees upon in a meeting and are usually assigned to one person. They are often part of meeting minutes, and are related to dialogue acts and discourse structure. In this work, software was created to annotate meeting transcripts. 10 meetings from the ICSI corpus were annotated, which were used together with the AIMU annotations to train a classifier on the task of action item detection. It was investigated to what extent the features lemmatised words, POS tags, and named entities are suitable for classifying utterances containing action items.

# Contents

**1 Introduction 1**

1.1 Motivation . . . . . . . . . . . . . . . . . . . . . . . . . 2
1.2 Objective . . . . . . . . . . . . . . . . . . . . . . . . . 2
1.3 Data and Code . . . . . . . . . . . . . . . . . . . . . . . 2

**2 Theory 3**

2.1 Textual features used . . . . . . . . . . . . . . . . . . . . 3
2.2 Spontaneous Speech . . . . . . . . . . . . . . . . . . . . 4
2.3 Dialogue Acts in Multi Party Meetings . . . . . . . . . . 5
2.4 Action Items . . . . . . . . . . . . . . . . . . . . . . . . 7
2.5 Related Work . . . . . . . . . . . . . . . . . . . . . . . . 8

**3 Methodology 10**

3.1 Data Sets Used . . . . . . . . . . . . . . . . . . . . . . . 10
3.1.1 ICSI Meeting Corpus . . . . . . . . . . . . . . . . . 10
3.1.2 AIMU Annotations . . . . . . . . . . . . . . . . . . 11
3.2 Corpus Annotation Tool . . . . . . . . . . . . . . . . . . 12
3.2.1 Data Model . . . . . . . . . . . . . . . . . . . . . . 13
3.2.2 Data Export Format . . . . . . . . . . . . . . . . . 15
3.3 Data processing pipeline . . . . . . . . . . . . . . . . . . 16
3.3.1 Feature Extraction . . . . . . . . . . . . . . . . . . 16
3.4 Classification Model . . . . . . . . . . . . . . . . . . . . 17

**4 Results 18**

4.1 Distribution of Features . . . . . . . . . . . . . . . . . . 18
4.1.1 Lemmatised words . . . . . . . . . . . . . . . . . . 19
4.1.2 POS-tags . . . . . . . . . . . . . . . . . . . . . . . 19
4.1.3 Named Entities . . . . . . . . . . . . . . . . . . . . 20
4.2 Experiments . . . . . . . . . . . . . . . . . . . . . . . . . 21
4.2.1 Original Distribution of Classes . . . . . . . . . . . 21
4.2.2 Undersampled Negative Class . . . . . . . . . . . . 22

**5 Discussion 24**

**6 Conclusion 26**

# 1 Introduction

Meetings are ubiquitous in today's professional life, whether in knowledge work to coordinate research efforts and measure their progress or in the corporate sector to evaluate resource planning and project progress. The documentation of decisions is of central importance, for several reasons, as shall be explained below.

**Meeting minutes** usually are written documents that summarise topics discussed, enlist participants present and serve as a record of the decisions made during the meeting and the next steps planned to carry out. Often there is a dedicated person responsible for taking such meeting minutes. In other cases, the project manager might take on this task himself.

Minutes do serve multiple purposes. Firstly they are a record the participants can refer to at a later point in time. Secondly, people concerned with the decision made but unable to attend themselves might use them as sources of information. Finally, in some cases, such minutes might even serve as a binding document to the decisions made.

**Action items** are often part of meeting minutes. Action items are similar to to-dos. They can be described as a task to be carried out at a specific time in the future, like: "Send an email to supplier XY, to ask about current price-lists." It naturally follows that such items need to be agreed upon by either the majority of the participants or the person responsible for making such decisions, e.g., a project manager. In some cases, they are listed in a separate section of a meeting minute to reference them quickly.

**Automatic Meeting Minute Generation** deals with the task of automatically detecting and recording all of the aspects of meeting minutes mentioned above, namely summarising of what has been discussed and by whom, enlisting of both: decisions made as well as action items agreed upon.

**Action Item Detection**: The following work deals with one aspect of automatic meeting minute generation, namely the automatic detection of action items in meetings, which are an essential part of most meeting minutes.

## 1.1 Motivation

The advantages of a system for automating the creation of meeting minutes are, on the one hand, to assist the person tasked with writing minutes, which can lead to potential improvements in meeting minutes, and, on the other hand, to make it possible to provide meeting minutes in settings where otherwise no such documents would have been created. Furthermore, automatically generated minutes can easily be created afterward, if needed, without the time and effort associated with manual post-processing, provided a recording of the respective meeting exists. Finally, the recognition and extraction of action items is the first step towards a system that can assign to-dos to team members automatically.

There are no publicly available systems or frameworks that allow to automatically generate meeting minutes that encompass automatic detection and extraction of action items to the author's best knowledge.

## 1.2 Objective

As can be seen in section 2.5, there are not many annotated corpora yet to train machine learning models for the task of action item detection. This paper aims to contribute by presenting a tool developed in the course of this work, which should facilitate the extension of existing corpora with annotations for action items and add action item annotations for 10 additional meetings of the ICSI corpus. Furthermore, the relevance of different textual features such as part-of-speech tags or named entities in the context of action item detection will be studied.

## 1.3 Data and Code

The following link points to a repository on Github, containing annotations created i.e. converted, the code for frontend and backend of the annotation software as well as the code used to carry out the experiments and generate the plots: `https://github.com/flurin-g/ActionAnnotator.git`

# 2 Theory

This section will first describe the textual features used in the experiments and the evaluation, then introduce the topics of spontaneous speech, dialogue acts, and, building on this, action items, before giving a brief overview of the existing literature.

## 2.1 Textual features used

Jurafski explains that parts of speech tags (POS) and named entities can help infer the meaning and structure of sentences. The related task of assigning a label to each word of an utterance is called sequence labelling and comprises both: POS-tags, as well as named entities [1].

**Lemmatised Words** Lemmatisation is a form of text normalisation, which aims at finding the root form of a word, from which it can differ in multiple ways. In other words, lemmatisation is the task of determining the common root of two words [1]. For instance, a verb can be used in different tenses, e.g., "be" as in "was" and "been" or it could be conjugated as in "am", "are" and "is", yet the root of all these words is "be", the same goes for inflected forms. By replacing all occurrences of words with their root, we can reduce the vocabulary size, which can help with summary statistics, i.e., word counts or bag-of-words approaches such as tf-idf vectors, since reducing the vocabulary size helps to reduce the dimensionality of the respective document vectors.

**Parts of Speech Tags (POS)** labelling each word in a sequence with their respective POS-tag means identifying the grammatical category of each word, for instance, verb, adjective, or noun. Parts of speech can belong to one of two categories, either open class or closed class. Closed class words are determiners, conjunctions, pronouns, prepositions, and auxiliary/modal verbs. These are more or less fixed, in the sense that rarely new words are added to them, contrary to open classes that comprise nouns and main verbs, where new words are frequently added. [1]

**Named Entities** One of the possible parts of speech a word can have, is NNP which indicates a proper noun, a proper noun can comprise a single word like "Paris", or it can comprise multiple words such as "Empire State Building". Regardless of the number of components, a proper noun always refers to a specific entity: a person, location, organisation, or specific time. Proper nouns, in turn, are closely related to the concept of named entities and the associated activity of named entity recognition. What makes named entity recognition (NER) a non-trivial task is the complexity of, on the one hand, identifying what entity is being referred to, e.g., does ”Zurich” refer to a city or an insurance company? and on the other hand, identifying its exact boundaries, when it comprises multiple words. The most common named entities are, according to Jurafski, PER (person), LOC (location), ORG (organization), and GPE (geopolitical entity), and are often extended to temporal expressions such as dates and times [1].

## 2.2 Spontaneous Speech

Today’s speech recognition systems achieve high recognition accuracies when applied to scripted speech. However, Nakamura et al. emphasize that spontaneous speech is fundamentally different acoustically as well as linguistically [2].

Ward et al. [3] describe 5 factors that are characteristic of spontaneous speech and can complicate downstream tasks, which they describe as follows:

- **Filled pauses:** Sounds uttered by the speaker that do not represent words
- **Restarts:** When a speaker interrupts a word or phrase, and then starts over, where the original word or phrase may be complete or interrupted.
- **Interjections:** Ward does not only mean interjections in the classical sense like ”wow”, but also extraneous phrases like ”on line thirty, I guess it is”.
- **Unknown or mispronounced words:** Strictly speaking, this is not a specific feature of spontaneous speech but an inadequacy in acoustic transmission.
- **Ellipsis:** When a speaker omits words or phrases that can be inferred from the context, such as ”Should I call you, or you me” instead of ”Should I call you, or should you call me”.
- **Ungrammatical constructions:** Sentence constructions which have an unusual order of their constituents, Ward gives as an

example the sentence: ”to the utilities cell add fifty dollars” [3]. Alternatively, grammatically incorrect formulations.

Shriberg et al. point out that positive aspects of spontaneous speech are that it ”... requires no special training, is remarkably efficient, imposes minimal cognitive load, and carries a wealth of information at multiple levels.” [4] yet poses special challenges to downstream tasks which Shriberg et al. categorise as follows [4]:

**Hidden punctuation** punctuations are not readily apparent in spoken language, since pauses during speech alone are not a good indicator of punctuations, and as pointed in [4], dialogue act modeling in particular benefits from correct punctuations.

**Disfluencies** As already described by Ward [3], disfluencies comprise filled pauses and repetitions. Shriberg et al. state that these affect up to a third of all utterances in many corpora, but they also note that they help to understand cognitive aspects of speech and interaction better.

**Realistic turn taking** Shriberg explains that speakers in dialogues do not take turns sequentially but anticipate the end of a sentence, and often start speaking before the previous speaker finishes his sentence, resulting in frequent overlap in spontaneous speech, which can lead to problems in the classification of dialog acts.

**Emotions of the speaker** Emotions are often conveyed by acoustic or prosodic features and cannot be represented by syntactic features.

## 2.3 Dialogue Acts in Multi Party Meetings

As explained in section 2.2 spontaneous speech introduces intricacies that are usually not present in written language, another layer of complexity regarding automatic information extraction is added when dealing with multi-party dialogues, and a means to structure such conversations are dialogue acts.

Dialogue acts can be ”... thought of as a tag set that classifies utterances according to a combination of pragmatic, semantic, and syntactic criteria.” [5]. Different standards to classify dialogue acts have been proposed, such as the AMI DA [6] or the DAMSL architecture [7]. The task of dialogue act recognition then is concerned with the concrete function an utterance has in a dialogue, which could be, for instance, a question, a suggestion, an offer, and so on, as McTear explains in [8]. Webb et al. emphasize that those dialogue acts represent the communicative intention behind an utterance [9], which, as Ang et al. note, is useful in a range of applications like determining to whom an utterance was addressed and whether participants agreed or disagreed on a specific topic [10]. Furthermore, dialogue acts are the cornerstone for explicitly organising a conversation into individual functional blocks, which in turn provides the basis for the discourse structure [8]. In addition to "Statements and Opinions", "Questions" and "Answers and Agreements", Stolcke mentions two other important dialogue acts which will be described in more detail below:

**Backchannels:** These are utterances whose intention is to encourage the speaker to continue, they are said to have a discourse-structuring role, and also help to recognise utterance boundaries [5].

**Turn Exits and Abandoned Utterances:** An abandoned utterance is one where a speaker stops in the middle and starts a new utterance without finishing the previous one. A turn exit is similar to an abandoned utterance in that the speaker also abandons it, but the purpose is to transfer the speaker role to someone else, and it is reported that a turn exit often includes the words "so" and "or" [5].

It is worth mentioning the connection between speaker-turns and utterances, because a speaker-turn can extend over several utterances, and a single utterance can extend over several speaker-turns, which would be the case with backchannelling of another speaker. The criterion for determining the boundaries of an utterance is in both cases whether a self-contained dialogue act is present [5].

Figure 1 shows dialogue act annotations for a transcript from the switchboard corpus. The 10th line shows an example of a backchannel uttered by speaker B, namely the word "Yeah", which, as described above, serves as an encouraging statement for speaker A to continue. Fernàndez et al. explain that the identification of the main conversational units in a decision-making process helps to identify regions where decisions are made [11], this, in turn, explains the concrete connection between dialogue acts and action items and motivates the study of dialogue acts as a means of improving the recognition of action-items.

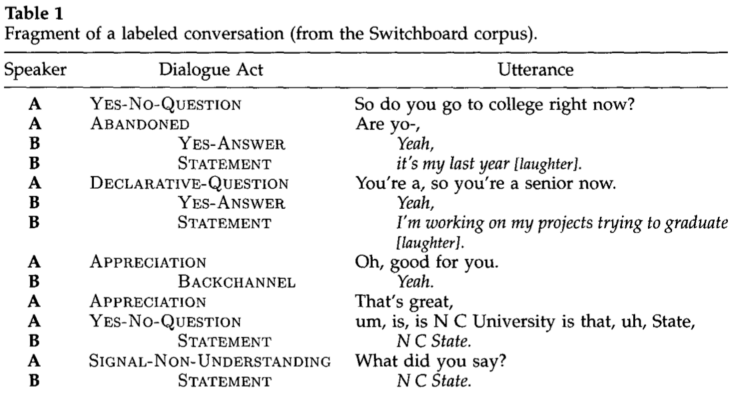


**Table 1**
Fragment of a labeled conversation (from the Switchboard corpus).

| Speaker | Dialogue Act | Utterance |
|---|---|---|
| **A** | Yes-No-Question | So do you go to college right now? |
| **A** | Abandoned | Are yo-, |
| **B** | Yes-Answer | *Yeah,* |
| **B** | Statement | *it's my last year [laughter].* |
| **A** | Declarative-Question | You're a, so you're a senior now. |
| **B** | Yes-Answer | *Yeah,* |
| **B** | Statement | *I'm working on my projects trying to graduate [laughter].* |
| **A** | Appreciation | Oh, good for you. |
| **B** | Backchannel | *Yeah.* |
| **A** | Appreciation | That's great, |
| **A** | Yes-No-Question | um, is, is N C University is that, uh, State, |
| **B** | Statement | *N C State.* |
| **A** | Signal-Non-Understanding | What did you say? |
| **B** | Statement | *N C State.* |

Figure 1: Transcript with dialogue act annotations taken from [5]

## 2.4 Action Items

Action items often are group decisions where the responsibility of a group is transferred to an individual, and they are thus an important element in the problem-solving process of organisations [12]. Action items are usually not formulated in one utterance by one speaker but comprise several utterances by different speakers [13]. Yang et al. describe 4 different classes of utterances, which have very different features and describe different aspects of action items [13]:

- **Description:** A description of the task to be executed as part of the action item.
- **Owner:** The individual responsible for executing the task, as mentioned earlier, this person is often determined in the context of a group decision.
- **Timeframe:** A time by which the task should be completed or a deadline.
- **Agreement:** Agreement of the other group members whether this action item should be carried out.

## 2.5 Related Work

The release of the ICSI corpus provided a multi-party meeting corpus that facilitated research for many topics regarding meeting understanding, one of which is the automatic detection and extraction of action items.

In [12] Purver et al. investigate automatic action item detection and propose a hierarchical annotation scheme to not just label but also categorize utterances that contain information relevant to an action item into 1 of 4 distinct categories. This annotation scheme is called AIDA. They report having annotated 65 meetings from the ICSI as well as the ISL meeting corpus. Their research showed promising results, but they also point out that more annotations are needed to generate train and test sets of sufficient size.

Morgan et al. also addressed automatic action item detection in [14] using the ICSI corpus, but with the annotations provided by Gruenstein et al. [15], reduced to a boolean response variable. In addition to lexical features (word uni-grams and bi-grams), they used POS tags, temporal features, dialog acts, and prosodic features. The classification was performed using a maximum entropy model. It is interesting to note that they report that their model achieves the best results with a combination of all features except syntactic features and dialogue acts.

Murray also deals with action item recognition in [16], but he uses transcripts from the AMI corpus. The annotations were created by having annotators write abstractive summaries, with several sections, one of which was a listing of action items. Similar to Morgan, they report the usage of several categories of features comprising prosodic and lexical features as well as dialogue acts. As a model, they chose a logistic regression classifier. Interestingly they note that the most effective features are cue words obtained by finding words frequent in meeting abstracts but less frequent in the transcripts themselves.

Similar to Purver in [12], Frampton et al. describe in [17] to have also used the ICSI meeting corpus, as well as the hierarchical annotations developed by Purver in [12]. The features Frampton adopted are similar to Purver's, but they also adopted the dialogue act annotations from the MRDA corpus [4]. In their experiments, they compared the performance of a HMM with that of an SVM, each using a classifier per AIDA category. They highlight that while the HMM gives good results in terms of precision, it performs significantly worse in terms of recall.

Yang et al. describe an interesting approach in [13], where they compare the dialogue acts from the MRDA corpus [4] with the action annotations from Purver [12]. They investigate how these dialogue acts are related to the action item annotations w.r.t. recall, precision and f-score, and find that the dialogue acts for commitment, suggestion, and command, i.e., action motivators, are closely related to action items.

A task that is closely related to action item detection is that of actionable item detection. Chen et al. describe their relationship as follows: ”action items are considered as a sub-group of actionable items that can be actionable in the form of reminders or to do lists.” [18]. The goal of Chen et al., as described in [18] is to develop approaches that facilitate the development of a meeting assistant that can assist meeting participants in real-time with functions such as creating to-dos, entering calendar items, or opening e-mails. For this purpose, they also use the ICSI corpus [12], adding annotations that indicate actionable items. To carry out annotations, they propose a scheme that includes 10 categories, of which only some are directly indicative of action items. To classify utterances into the different categories of actionable items, they also use an SVM and provide intent and utterance embeddings obtained by utilising convolutional deep structured semantic models (CDSSM).

The aim of Tran et al. [19] is to extract decision elements from meeting transcripts; therefore, they target sequences of text that are directly related to decision processes, which they obtain by labeling sequences of words on a sub-utterance level. Their annotation process comprises 3 steps, combining annotations from domain experts with annotations obtained through crowdsourcing and having domain experts revising those annotations again. Similar to [14] they used Maximum Entropy Models, but in addition, they also employed conditional random fields [20].

# 3 Methodology

This section first describes the corpora used, then introduces the annotation software developed for this work and the data model used for persistence. Next, the data processing pipeline is presented, which includes the pre-processing and the process of feature extraction. After that, the classification model and its structure are explained.

## 3.1 Data Sets Used

The datasets used in this paper are linked in the sense that the ICSI Meeting corpus contains all transcripts but not the action item annotations. There is, however, another related corpus called MRDA that contains some dialogue acts such as "suggestion", "disagreement", or "joke" [21], which are not directly applicable to the use-case of action-item detection. The AIMU Corpus, on the other hand, does not contain any transcriptions of meetings but uses those provided by the ICSI Meeting Corpus and adds annotations to those, which were used as a starting point in the present work. Common to both corpora is that they both contain manual annotations.

### 3.1.1 ICSI Meeting Corpus

As described by Janin et al. in [21] the ICSI Meeting Corpus consists of 75 Meeting recordings recorded at the ICSI in Berkeley between the years 2000 and 2002, most of which were regularly scheduled meetings, meaning they are natural, unscripted meetings as opposed to synthetic meetings. The reported average duration of meetings is slightly under 1 hour, with 3-10 participants each, averaging 6 participants. They report that some of the speakers were native English speakers while others were not [21]. One can infer the individual topic/purpose of the meeting by looking at the filename. There are 10 distinct categories of meetings, with titles such as "Database issues meeting", "Even Deeper Understanding weekly meeting", or " Meeting Recorder weekly meeting".

In addition to lexical features, the transcript also includes interjections and pause fillers such as "uhm", "hmm", "yeah", and "ok", which often serve the purpose of backchannels. Also, repetitions of words are transcribed as is. A peculiarity of the transcription is that successive utterances need not be from different speakers, i.e., there might be three utterances in series that all belong to the same speaker, which might be because the authors decided to segment utterances according to dialogue acts rather than speaker turns.

### 3.1.2 AIMU Annotations

The AIMU annotations emerged in the course of investigating approaches to identifying actionable items in meetings which are described in more detail by Chen et al. in [18]. According to them, actionable items comprise discussions on scheduling, emails, action items, and search. ”The actionable items annotations are performed on a subset of the ICSI meeting corpus” [18]. The resulting corpus comprises annotations for 22 meetings, where Chen et al. would limit the interviews chosen to 3 of the original 10 categories defined in [21]. Those are:

- **Bed**: Even Deeper Understanding meetings
- **Bmr** Meeting Recorder meetings
- **Bro** Robustness meetings

The annotations of Chen et al. are divided into 5 domains, each of which has 2 subcategories, namely intent or argument, resulting in a total of 10 possible annotations.

They report an average inter-annotator agreement of 0.644 about whether an utterance includes an actionable item [18]. They further mention that ”here are total 318 turns annotated with actionable items, which account for about 1.5% of all of the turns” [18].

```
O_K. Have a great meeting. <create_single_reminder>
I’ll - I’ll come back up <start_time>in about an
hour</start_time> and <reminder_text>check and see
if you’re still meeting</reminder_text> .
</create_single_reminder>
```

Listing 1: Annotated Utterance from AIMU Corpus [18]

Listing 1 shows a single annotated utterance from the AIMU corpus [18], as can be seen, these annotations need not span a complete utterance and can be nested as well.

## 3.2 Corpus Annotation Tool

The CorpusAnnotater is a web application, meaning it consists of a frontend implemented as a single page application (SPA) and a backend, which communicate over a REST API. The aim behind the development of the application was to provide a tool that offers an intuitive interface that allows to quickly annotate existing corpora with labels that can be used to train machine learning models on the task of action item detection.

The tool is intended to be used on transcripts of multi-party meetings, where those transcripts are segmented into utterances per speaker, as is the case for the ICSI corpus. Thus, it does not matter whether there are multiple successive utterances by the same speaker or if an utterance comprises a speaker turn.

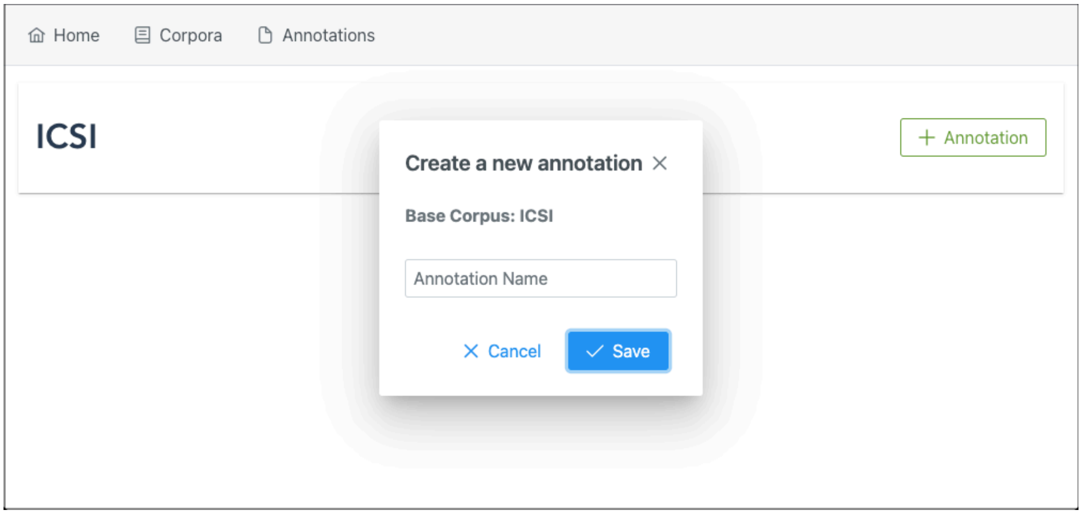
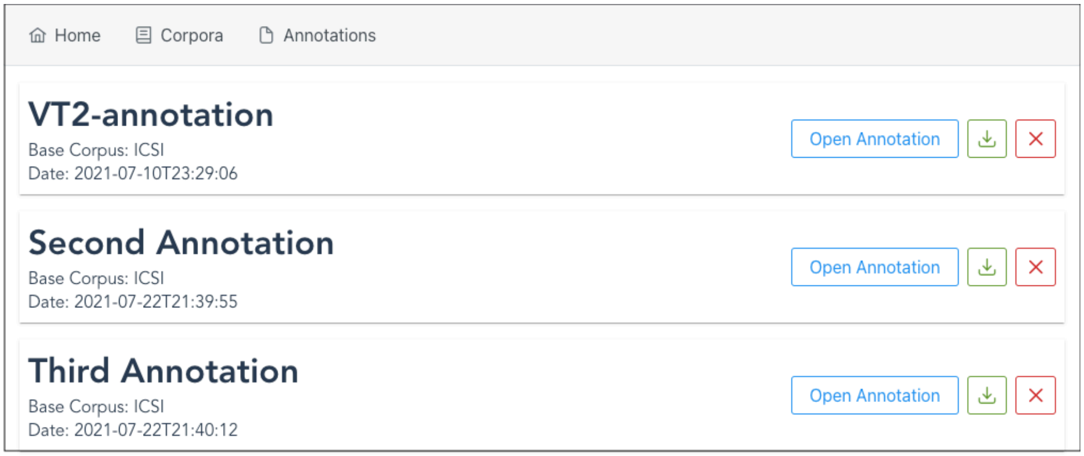


Figure 2: Corpus view, with new annotation dialogue

**Corpus view**: The architecture of the software was implemented in such a way that it should be possible to extend it with other corpora. Figure 2 shows the corpus overview, which lists each of the integrated corpora. It also shows the dialogue the user is presented with when clicking the button to add an annotation.

**Annotation view**: The annotation view as shown in figure 3 lists all annotations created by a user. Furthermore, it shows the base corpus meaning the corpus the annotation is built on and the creation date of the annotation. It provides controls to delete an annotation as



VT2-annotation
Base Corpus: ICSI
Date: 2021-07-10T23:29:06

Second Annotation
Base Corpus: ICSI
Date: 2021-07-22T21:39:55

Third Annotation
Base Corpus: ICSI
Date: 2021-07-22T21:40:12

Figure 3: Annotation view, overview of created annotations

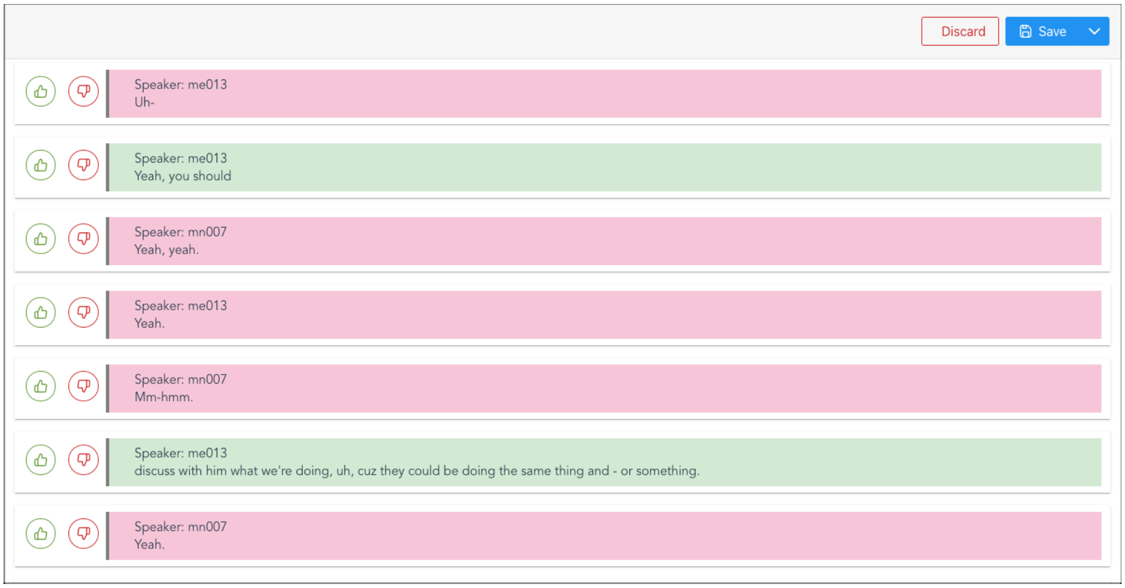
well as to download it. Initiating a download causes the software to combine the annotations with the associated transcript and transform them into JSON file for all transcripts contained in a corpus. When a user clicks the ”Open Annotation” button, the annotation editor opens up.

**Annotation Editor**: The annotation editor, as shown in figure 4, provides the interface to annotate all the utterances contained in a transcript quickly. By clicking either the thumbs-up or thumbs-down button, an utterance is either labeled as ”isActionItem”: ”yes” or ”no”. The initial state of the ”isActionItem” field is ”maybe”, which would allow providing additional information to the user as to whether a transcript still has missing annotations or how many percent of the corpus have already been annotated. If a user decides not to store the changes made, he can usher the discard button, and if he wants to persist his changes, he can usher the save button. No undo-/redo-functionality is implemented, but the data model used facilitates implementing this functionality at a later point in time.

### 3.2.1 Data Model

To persist a user's annotations, ActionAnnotater uses a document-store in the backend, whose data model is depicted in figure 5.

The data model is structured so that annotations are handled as



Speaker: me013
Uh-

Speaker: me013
Yeah, you should

Speaker: mn007
Yeah, yeah.

Speaker: me013
Yeah.

Speaker: mn007
Mm-hmm.

Speaker: me013
discuss with him what we're doing, uh, cuz they could be doing the same thing and - or something.

Speaker: mn007
Yeah.

Figure 4: Annotation editor, label utterances as action item or not

distinct entities linked to the transcript to be annotated, which avoids the storage of duplicate data and enables faster updates with the frontend since only the annotations themselves are added/altered/removed in the annotation process. When exporting a corpus, the software provides the necessary functionality to combine the annotations with the utterances of the transcripts.

As can be seen in figure 5, a corpus comprises 1...n transcripts, each of which contains 1...n utterances. Each annotation links to a single corpus referred to as its base corpus. Similarly, each transcript annotation links to a single transcript, which it refers to through a transcriptId.

It remains to be mentioned that corpora and transcripts are maintained as independent tables, whereas utterances are maintained in an embedded array within the transcript, likewise for the utterance annotations of a transcript annotation. The embedded array will facilitate versioning (undo/redo) of the transcript annotations and also simplifies the schema.

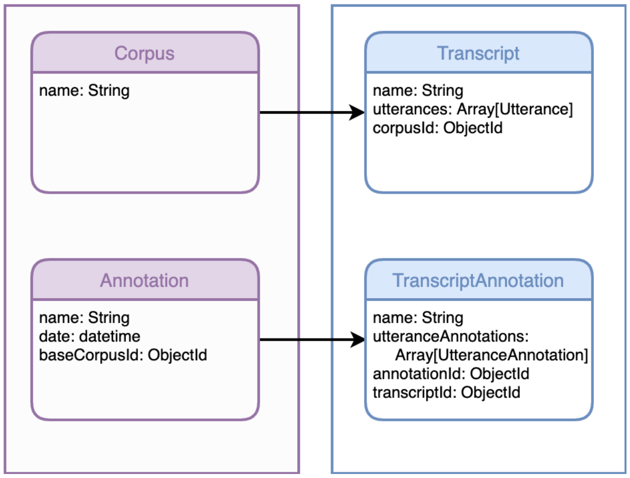


Figure 5: Data model used to store annotations

```
[
    {
        "speaker": "me012",
        "text": "Yeah Ive have never handled them",
        "isActionItem": "no"
    },
    {
        "speaker": "me003",
        "text": "Goats eat cans to my understanding",
        "isActionItem": "no"
    }
]
```

Listing 2: Format of JSON-Export

### 3.2.2 Data Export Format

The ActionAnnotator software exports documents in JSON format. The export format is JSON because it should be as easy as possible to import the resulting documents into a subsequent application for further processing. As can be seen in listing 2, an exported transcript consists mainly of an array of objects, which have attributes for name, the text of utterance, and a label that can have the values ”yes”, ”no”, and ”maybe”.

## 3.3 Data processing pipeline

In the course of the present work, the ActionAnnotator software was used to annotate 10 meeting annotations of the ICSI corpus, namely the following ones:

- Bro003
- Bro005
- Bro007
- Bro010
- Bro012
- Bro013
- Bro015
- Bro016
- Bro017
- Bro028

To keep the annotation process as streamlined as possible, the annotation scheme of AIMU was not adopted. Instead only the label described in section 3.2.2 was used. Therefore, to combine the AIMU annotations with the additionally annotated 10 meetings, the AIMU annotations had to be converted.

The conversion included two aspects, firstly the AIMU annotations in *.trans format were converted to JSON, and secondly, the annotations in sub-utterance level were converted to utterance level annotations. Not all annotations used in [18] indicate an action item. More precisely, out of the 10 categories defined under [18], only the presence of the 3 tags in listing 3 were considered to set the ”isActionItem” label of an utterance to ”yes”:

```
"<send_email>"
"<create_calendar_entry>"
"<create_single_reminder>"
```

Listing 3: Tags indicative of an action item

### 3.3.1 Feature Extraction

The raw data for the feature extraction is in JSON format, stored as shown in listing 2. All files are loaded from disk, and the respective features are computed and added to a JSON object that is part of an array that again makes up a transcript. For each transcript, i.e., file, the resulting array is written back to disk. The features comprise:

- Lemmatised word tokens
- POS Tag
- Named Entities

Besides lowercasing and stop-word removal, non-alpha characters were removed. The remaining words were then stored as-is. Finally, the additional features were extracted using Spacy's large English language model, where the authors report that this model was trained on OntoNotes5 and WordNet [22].

**Lemmatised words and POS-tags**: Each tokenised word, along with the corresponding lemmatised token as well as the POS-tag, make up an object that is part of the array of objects that is embedded in the object comprising a single utterance.

**Named entities:** For the utterances that did not contain named entities, an object was created whose entity field contained the text ”None” and whose text field was empty. If a named entity was detected, its identifier was entered into the entity field, and the text was stored in its own ”Text” field. The entities are kept as an array of objects per utterance because a 1-to-1 mapping to tokens is not possible. Furthermore, unlike a POS tag, an entity can contain several words, e.g., ”Empire State Building”.

The same features were used for both the summary statistics and the classifier.

## 3.4 Classification Model

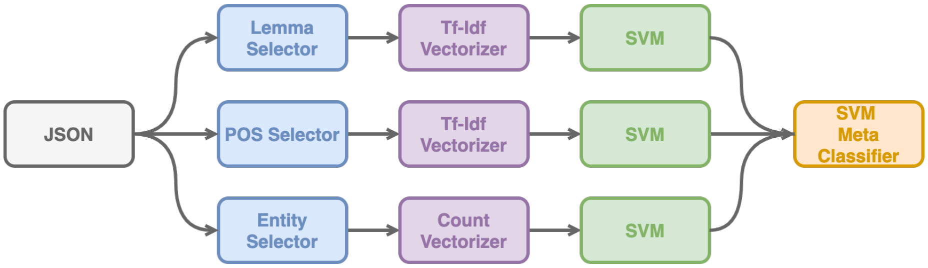


Figure 6: Architecture of the classification model

Figure 6 depicts the architecture of the classification model. As a first step, shown in the grey box, the already computed features are loaded from the disk into a JSON object, which serves as input for the classification model. After that, the classifier is divided into three parallel paths. In each of the three paths, one of the three features (lemmatised token, POS-tag, and named entity) is extracted by a feature-selector transformer, shown here as a blue box. For lemmatised tokens as well as POS-tags, tf-idf document vectors are created per utterance. For the lemmatised tokens, word unigrams, bigrams, and trigrams are created for the POS-tags bigrams and trigrams, respectively. For the named entities, a simple count-vectoriser is used. All 3 paths are then fed into their individual SVM classifier (namely the SVC implementation from scikit-learn [23]). The output of each classifier is then fed into another SVM instance of the same type that acts as a meta classifier, where the outputs of the individual SVMs are stacked and used by the meta classifier to make the final prediction as implemented by scikit-learn's stacked generalisation classifier [23].

# 4 Results

In the following, the features discussed are examined from two different perspectives: firstly, in section 4.1, count statistics of the different features are used to examine differences between utterances containing action items and utterances that do not contain action items. Secondly, using the classifier described in section 3.4, experiments are performed to evaluate the impact of the different features on the performance of the classifier.

## 4.1 Distribution of Features

In the following, word-count diagrams for the 3 features lemmatised tokens, POS-tags, and named entities are shown. Because of the substantial imbalance in favor of utterances that are not action-items, the diagrams which consider the features independent of class strongly overlap with those for the class ”no action-item”. Therefore, these diagrams are not shown, and instead, the diagrams for ”no action-item” and ”is action-item” are displayed per feature.

### 4.1.1 Lemmatised words

What is evident in the word count for lemmatiesd words is that out of the 12 words juxtaposed, 7 occur in both classes, although not in the same rank. It seems interesting that two words that could indicate a temporal frame of reference, namely ”start” and ”week”, only occur in the ”is action-item” class. Likewise, ”talk” occurs only in the positive class.

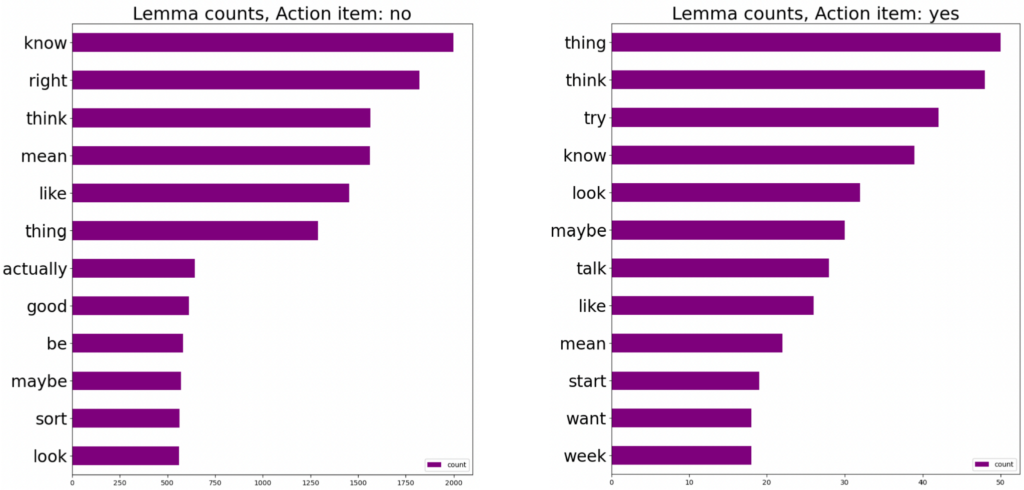


Figure 7: Distribution of lemmatised words for action items

### 4.1.2 POS-tags

The first thing that is noticeable when comparing the diagrams of the POS tags in figure 8 is that there are proportionally more nouns in the utterances that do not contain action items. The diagram describing the action items shows that the two most frequent POS tags have almost the same number of counts, while the diagram for no action items shows a noticeable drop from the most frequent to the second most frequent POS tag. The 3rd to 7th most frequent counts are the same for both classes: adjectives, proper nouns, adverbs, interjunctions, and subordinating conjunction.

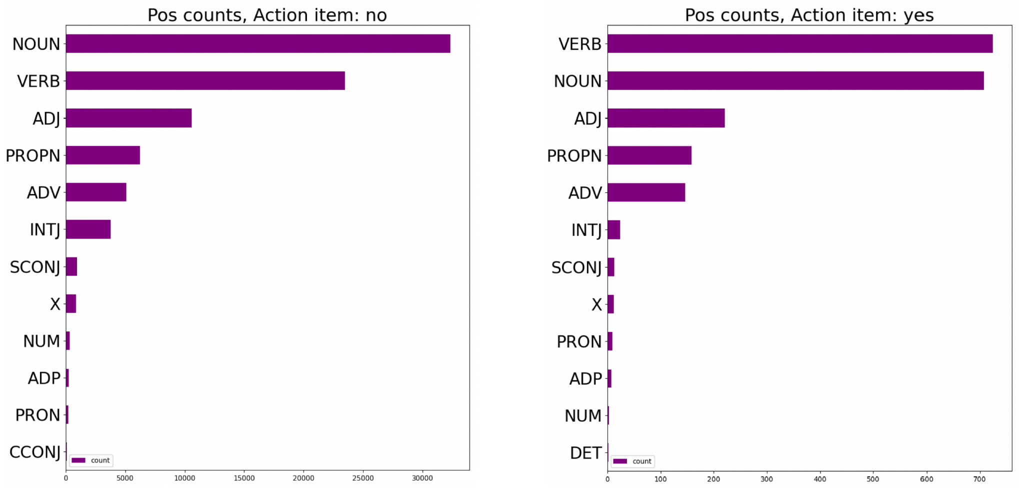


Figure 8: Distribution of part-of-speech tags for action items

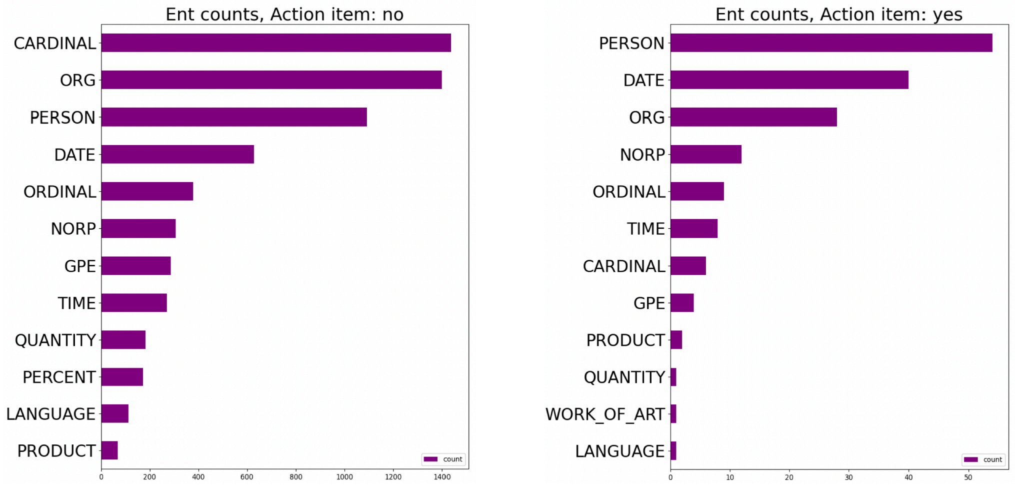


Figure 9: Distribution of named entities for action items

### 4.1.3 Named Entities

In figure 9 the diagrams for the counts of the named entities are compared. What stands out in the diagram of the action items is that the most common entity is ”Person”, followed by ”Date”. The diagram for the utterances which do not contain action items shows that the entity ”Person” is ranked 3rd and the entity ”Date” is ranked 4th.

## 4.2 Experiments

In the experiments, the performance of the stacked generalization classifier using different features and differently distributed data was evaluated.

The complete number of utterances after pre-processing is 19'667. However, there is a substantial imbalance between the class of utterances containing action items, which comprises just 274 utterances or 1.39%, compared to 19'393 utterances or 98.61% for the class of utterances not containing action items. The same test set was always used for all experiments. The ratio of train/test-split is 0.8 to 0.2, resulting in a support of 55 samples for the positive class, that is, utterances containing action items.

### 4.2.1 Original Distribution of Classes

| **Features** | **f1 cross validated** | **f1 on test set** |
|---|---:|---:|
| Lemma | 0.000 | 0.00 |
| Lemma + Pos | 0.164 | 0.27 |
| Lemma + Pos + Entity | 0.235 | 0.23 |

Table 1: Results for f1 score - cross validated same distribution as test set

In table 1 the f1-scores for the stacked generalisation classifier using different features are shown. The second column shows the f1-scores of the 5-fold cross-validated evaluation. The third column shows the results on the test set. The f1-score for the classifier, which uses only n-grams of the lemmatised words as features, is 0, which is since for the positive class, not a single utterance was detected, i.e., the recall is 0 as well. A recognisable increase can be seen when POS-tag n-grams are added to the lemmatised word n-grams as features, leading to an f1-score of 0.164 for the cross-validated score, and an f1-score of 0.27 on the test set, respectively.

### 4.2.2 Undersampled Negative Class

In the experiments on the undersampled data, the number of samples in the positive class, i.e., the class of action items, was determined, and the same number of negative samples was then used accordingly.

| **Features** | **f1 cross validated** | **f1 on test set** |
|---|---:|---:|
| Lemma | 0.835 | 0.45 |
| Lemma + Pos | 0.855 | 0.54 |
| Lemma + Pos + Entity | 0.877 | 0.55 |

Table 2: Results for f1 score - cross validated undersampled negative class

The classifier was again trained in the same manner as described in section 4.2.1 with the different features. In the second column of table 2 one can see cross-validated f1-scores on the undersampled training data. Again, similar to 4.2.1, the classifier that uses only lemmatised words as features has a lower f1-score of 0.835 than the classifier that additionally uses POS-tag n-grams with a score of 0.855, and again the classifier that combines all 3 features, i.e., lemmatised word n-grams, POS-tag n-grams, and named entities has the highest f1-score on the cross validated results. What is interesting to note is that when comparing the f1-scores in the 3rd column of figure 2 with those from figure 1 the scores are noticeably higher, one should keep in mind that the same test set as in the experiments using all negative samples is used. However, in the first case, the classifier was trained on 19,667 utterances, whereas in the second case, it was trained on only 548 utterances, while still resulting in higher f1-scores than in the first case, namely 0.45, 0.54, and 0.55.

| **Features** | **accuracy** |
|---|---:|
| Lemma | 0.808 |
| Lemma + Pos | 0.851 |
| Lemma + Pos + Entity | 0.876 |

Table 3: Cross validated accuracy scores on undersampled negative class

Since, in the case of the experiments with the under-sampled negative class, a balanced ratio between utterances containing action items and those containing no action items exists, the values for accuracy can also be used here to assess the classifier's performance when using different features. Again, adding POS-tag n-grams to the lemmatised word n-grams leads to an improvement from 0.808 to 0.851. Somewhat less pronounced but still noticeable is the increase when adding named entities as features, with a resulting accuracy of 0.876, which can be seen in table 3.

# 5 Discussion

The discussion will be done mainly by looking at the features and their influence on the classification results. After that, the use of the data in the classification experiments will be considered in more detail.

**Lemmatised Words** What is notable about considering the most frequently lemmatised words, as shown in figure 7, is that no words with a temporal reference appear in the utterances that do not contain action-items, which is, however the case in the positive class, namely the two words "start" and "week". It should also be mentioned that, when evaluated on the training data set with the original distribution, the classifier was not able to recognise a single utterance that contained an action item. This may be due to the highly unbalanced data, but it is still surprising, as adding features has substantially increased the respective scores.

**POS Tags** What seems noteworthy is that there are no significant differences between the POS counts for the plots for positive and negative classes. Besides the inverted first and second ranks, yet the addition of POS n-grams as input features increased the f1-scores, i.e., the accuracy of the conducted experiments noticeably. A possible explanation for this might be that bi-grams and tri-grams allow the classifier to better model syntactic structures indicative of action items. The high number of proper nouns (over 5'000 for the negative class and 160 for the positive class, respectively) motivated the examination of named entities.

**Named Entities** What can be seen in figure 9 is the relatively higher number of the named entity "date" in the utterances containing an action item. Moreover, if we refer to the definition of action-items as give in section 2.4, we can see that a central criterion for it to qualify as such is its connection to a specific point in time. Thus, the higher number of the entity "date" in the counts for the positive class in combination with the increased f1-score, i.e., accuracy for the classifier including named entities as features, motivates further examination of whether there is a positive correlation between the presence of the named entity "date" and an utterance being an action-item.

**Results of the Classification Experiments** Most striking in the evaluation of the results of the experiments is undoubtedly the substantial difference between the results on the training data with the original distribution and those generated by undersampling the utterances that do not contain action items. Whereas the f1 score on the original distributed data remained below 0.3 points for all features, it rose above 0.8 points in the case of the undersampled training data. Even more surprising is the fact, that even on the test set better scores were achieved, when the classifier was trained on the much smaller undersampled data set.

# 6 Conclusion

The results of the experiments clearly show that action-item detection at the utterance level is possible with binary labels and using the features proposed in this paper, such as the presence of the named entity "date". In this paper, a contribution was made with the development of an annotation tool that facilitates the rapid annotation of transcripts with binary labels for action-item detection to support future efforts in this direction. In addition, the 22 AIMU annotations were converted to this binary format, and a further 10 meeting transcripts were annotated with binary labels. The 32 meeting transcripts are available in JSON format at the repository referenced in section 1.3. What is evident from the experiments in this paper is that the highly unbalanced distribution of classes in the data is a significant problem for the task of action-item detection, and the author believes that solving this problem is an essential step towards reliable action-item detection. In the present work, this has been tried with undersampling the dominant class and has shown promising results, but other approaches might be worth investigating. Furthermore, binary annotation schemas do not seem to be sufficient for the task of action-item recognition. Schemas are needed that allow differentiation of various dialogue acts, ideally adapted to the task of action-item recognition, in order to extract all information belonging to a given action item. Promising approaches here are certainly the hierarchical schema [12] proposed by Purver et al. or the one presented by Chen et al. [18] for the segmentation of actionable items. However, these require much more effort in the annotation of transcripts.

# References

[1] Jurafsky Daniel and Martin James H. Speech and language processing. daniel jurafsky & james h. martin., 2020.

[2] Masanobu Nakamura, Koji Iwano, and Sadaoki Furui. Differences between acoustic characteristics of spontaneous and read speech and their effects on speech recognition performance. *Computer Speech & Language*, 22(2):171–184, 2008.

[3] Wayne Ward. Understanding spontaneous speech: The phoenix system. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, pages 365–367. IEEE Computer Society, 1991.

[4] Elizabeth Shriberg, Raj Dhillon, Sonali Bhagat, Jeremy Ang, and Hannah Carvey. The icsi meeting recorder dialog act (mrda) corpus. Technical report, INTERNATIONAL COMPUTER SCIENCE INST BERKELEY CA, 2004.

[5] Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational linguistics*, 26(3):339–373, 2000.

[6] Daniel Gatica-Perez, L McCowan, Dong Zhang, and Samy Bengio. Detecting group interest-level in meetings. In *Proceedings.(ICASSP'05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, volume 1, pages I–489. IEEE, 2005.

[7] Mark G Core and James Allen. Coding dialogs with the damsl annotation scheme. In *AAAI fall symposium on communicative action in humans and machines*, volume 56, pages 28–35. Boston, MA, 1997.

[8] M McTear, Z Callejas, and D Griol. The conversational interface: Talking to smart devices: Springer international publishing. *Doi: https://doi. org/10.1007/978-3-319-32967-3*, 2016.

[9] Nick Webb, Mark Hepple, and Yorick Wilks. Dialogue act classification based on intra-utterance features. In *Proceedings of the AAAI Workshop on Spoken Language Understanding*, volume 4, page 5. Citeseer, 2005.

[10] Jeremy Ang, Yang Liu, and Elizabeth Shriberg. Automatic dialog act segmentation and classification in multiparty meetings. In *Proceedings.(ICASSP'05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, volume 1, pages I–1061. IEEE, 2005.

[11] Raquel Fernández, Matthew Frampton, Patrick Ehlen, Matthew Purver, and Stanley Peters. Modelling and detecting decisions in multi-party dialogue. In *Proceedings of the 9th SIGdial Workshop on Discourse and Dialogue*, pages 156–163, 2008.

[12] Matthew Purver, Patrick Ehlen, and John Niekrasz. Detecting action items in multi-party meetings: Annotation and initial experiments. In *International Workshop on Machine Learning for Multimodal Interaction*, pages 200–211. Springer, 2006.

[13] Fan Yang, Gokhan Tur, and Elizabeth Shriberg. Exploiting dialogue act tagging and prosodic information for action item identification. In *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4941–4944. IEEE, 2008.

[14] William Morgan, Pi-Chuan Chang, Surabhi Gupta, and Jason Brenier. Automatically detecting action items in audio meeting recordings. In *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue*, pages 96–103, 2006.

[15] Alexander Gruenstein, John Niekrasz, and Matthew Purver. Meeting structure annotation: Data and tools. In *6th SIGdial Workshop on Discourse and Dialogue*, 2005.

[16] Gabriel Murray and Steve Renals. Detecting action items in meetings. In *International Workshop on Machine Learning for Multimodal Interaction*, pages 208–213. Springer, 2008.

[17] Matthew Frampton, Raquel Fernández, Patrick Ehlen, Anish Adukuzhiyil, and Stanley Peters. Leveraging minimal user input to improve targeted extraction of action items. In *LONDIAL 2008 the 12th Workshop on the Semantics and Pragmatics of Dialogue*, page 108, 2008.

[18] Yun-Nung Chen and Dilek Hakkani-Tur. Aimu: Actionable items for meeting understanding. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 739–743, 2016.

[19] Tuan Tran, Francesca Bonin, Léa A Deleris, Debasis Ganguly, and Killian Levacher. Preparing a dataset for extracting decision elements from a meeting transcript corpus.

[20] John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

[21] Adam Janin, Jeremy Ang, Sonali Bhagat, Rajdip Dhillon, Jane Edwards, Javier Macias-Guarasa, Nelson Morgan, Barbara Peskin, Elizabeth Shriberg, Andreas Stolcke, et al. The icsi meeting project: Resources and research. In *Proceedings of the 2004 ICASSP NIST Meeting Recognition Workshop*, 2004.

[22] Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. spaCy: Industrial-strength Natural Language Processing in Python, 2020. URL: `https://doi.org/10.5281/zenodo.1212303`, `doi:10.5281/zenodo.1212303`.

[23] 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. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

# List of Figures

1 Transcript with dialogue act annotations taken from [5] 7
2 Corpus view, with new annotation dialogue . . . . . . 12
3 Annotation view, overview of created annotations . . . 13
4 Annotation editor, label utterances as action item or not 14
5 Data model used to store annotations . . . . . . . . . 15
6 Architecture of the classification model . . . . . . . . . 17
7 Distribution of lemmatised words for action items . . . 19
8 Distribution of part-of-speech tags for action items . . 20
9 Distribution of named entities for action items . . . . 20

# List of Tables

1 Results for f1 score - cross validated same distribution as test set . . . . . . . . . . . . . . . . . . . . . . . . . 21
2 Results for f1 score - cross validated undersampled negative class . . . . . . . . . . . . . . . . . . . . . . . . . 22
3 Cross validated accuracy scores on undersampled negative class . . . . . . . . . . . . . . . . . . . . . . . . . 23