# Automatic Classification of Non-Verbal Utterances in Japanese Spontaneous Speech


Interactive Systems Labs.

Universität Karlsruhe (TH)
Institut für Logik, Komplexität und Deduktionssysteme (ILKD)


ADVANCED
TELECOMMUNICATION
RESEARCH INSTITUTE
INTERNATIONAL

Studienarbeit

Cand. -Inform. Wenzel Svojanovsky

June 2004

Supervisors:
Prof. Dr. rer. nat. Alexander Waibel
Dipl. -Inform. Rainer Gruhn
Dipl. -Inform. Hartwig Holzapfel

**Abstract**

Spontaneous speech is more difficult to understand than read text. Spontaneous speech is expressive speech, so the interpretation of *what* is said is difficult as long as there is no information about *how* it is said.
Tokens like "un" in Japanese spontaneous speech and "mhm" in English spontaneous speech are critical for a communication. These *backchannels* carry information about the speaker's state of mind and control the flow of words. They have different meanings like "Yes" and "No".
The data for this project is Japanese everyday speech. The speaker is a Japanese female wearing a microphone and a Minidisk recorder whole day.

The goal is to achieve an automatic interpretation of these backchannels. Prosodic features based on F0 (pitch), duration, power, and glottal characteristics will be extracted from tokens and automatically clustered and classified into one of several speech act classes.

In this research different clustering algorithms are tested and discussed, also the relevance of the single features are analyzed.

Spontan gesprochene Sprache ist schwerer zu verstehen als vorgelesener Text, da diese Form der Sprache ausdrucksstärker ist. Deshalb ist das Verstehen von *was* gesagt wurde schwer, solang es keine Information darüber gibt *wie*, es gesagt wurde.
Wörter wie "un" im Japanischen und "mhm" im Englischen sind unabdingbar in spontan gesprochener Sprache. Diese *backchannels* tragen Information über die Ansichten des Sprechers und beeinflussen den Sprachfluss. Sie haben verschiedene Bedeutungen wie "Ja" und "Nein".
Bei den Daten für dieses Projekt handelt es sich um japanische Alltagssprache. Die Sprecherin ist eine Japanerin, die ein Mikrofon und einen Minidiskrecorder den ganzen Tag trug.

Das Ziel ist eine automatische Interpretation dieser *backchannels*. Prosodische Merkmale, beruhend auf F0 (Grundfrequenz), Laufzeit, Energie und Glottis–Ausprägung, werden aus den Äußerungen extrahiert und automatisch geclustered und zu einer Bedeutungsklasse zugeordnet.

In dieser Arbeit werden verschiedene Cluster–Algorithmen getestet und die Bedeutung der einzelnen Merkmale analysiert.

# Contents

# Chapter 1

# INTRODUCTION

This chapter will introduce the reader to the topic of this student project.

## 1.1 Motivation

The following dialog is a short section of the transcription en_4677 of the English CALLHOME database distributed by the LDC (Linguistic Data Consortium) [10].

[...]

**A:** "oh"

**B:** "so what"

**B:** "yeah"

**A:** "yeah"

**B:** "mhm"

**A:** "you there"

**B:** "yeah"

**A:** "okay"

**B:** "it's the twenty first I start on the thirty first [piano]"

**A:** "mhm"

**A:** "%hm I start September fifth"

**B:** "yeah so does middle school but"

**A:** "mhm"

**B:** "oh well exhale"

**A:** "oh well"

**A:** "too bad [[softly]]"

[...]

It is well known, that spontaneous speech is more difficult to understand than read text. This applies both to humans and machines. Indeed, the performance of an automatic speech recognition system drops substantially, when analyzing spontaneous speech.

The dialog in the example above shows that several utterances, especially "mhm" and "hm" occur frequently, with rather no semantic information left, when transcribed the way above. There is no way to derive the complete meaning from the transcription.

Since spontaneous speech is *expressive speech* the interpretation of *what* is said is impossible, as long as there is no information about *how* it is said.

In the future, human–machine communication will become a common action like programming the a video recorder or a humanoid robot by voice, or to speak to another person using an automatic translation system.

The semantic of the uttered words is critical to make this kind of communication possible at all. Without that, machines will understand our words, but not their meaning.

## 1.2 Topic

In this project, the filled pauses that are usually transcribed as "un" in Japanese speech are analyzed. These kinds of utterance shall hence be termed as *backchannel*. These backchannels play an important role in spontaneous speech and have different meanings like "Yes", "No", "Maybe", or as dialog flow control. For semantic analysis of spontaneous dialogues, the interpretation of these utterances is essential.
Therefore prosodic features will be extracted, automatically clustered, and classified. These features are explained in section 2.1.1 .
The data for this project was recorded for the ESP (Expressive Speech Processing) project by JST/CREST at ATR HIS Labs, Kyoto. More information about the data can be found in section 5.1.1
The goal is to achieve an automatic interpretation of the backchannel "un" to analyze the semantic context of a dialog. Furthermore it shall give information about the speaker's state of mind.

This approach is extended to another backchannel interpretation problem, the token "honma". Neither the size nor the labels of both sets are comparable. The same experiments will run on this additional data set to compare the quality of the clustering method.

## 1.3 Overview

This section gives a short overview of the structure of this student project.

First of all the fundamentals, which are necessary for the understanding of this approach, are explained in chapter 2. It shows the idea of prosody and clustering.
Following chapter 3 gives a short overview of the current state of the art. Various papers, reports and theses about filled pauses and prosody are summarized. This provides an insight into related work.
Chapter 4 about classification methods illustrates the project specific methods. The preparation of the data, experimental setup, classifier and evaluation is explained.
The experiments and the data are exposed in chapter 5. It also presents the results of the clustering.
In chapter 6 the single parameters of the experiment and their influence to the results are discussed.

# Chapter 2

# FUNDAMENTALS

This chapter gives a short overview of the terminology and technics which are used during this approach.

## 2.1 Prosody

This section explains the meaning and importance of prosody and discusses prosodic features.

### 2.1.1 What is Prosody?

According to Fujisaki [8], prosody has to be considered from two different aspects, the "concrete aspect"—defining prosody in physical term, and the "abstract aspect"—defining prosody as influence to linguistic structure.

**concrete aspect:** phenomena that involve the acoustic parameters of pitch, duration, and intensity

**abstract aspect:** phenomena that involve phonological organization at levels above the segment

Prosody in speech has both, measurable manifestations and underlying principles. Therefore the following definition is appropriate:

Prosody is a systematic organization of various linguistic units into an utterance or a coherent group of utterances in the process of speech production. Its realization involves both segmental features of speech, and serves to convey not only linguistic information, but also paralinguistic and non-linguistic information.

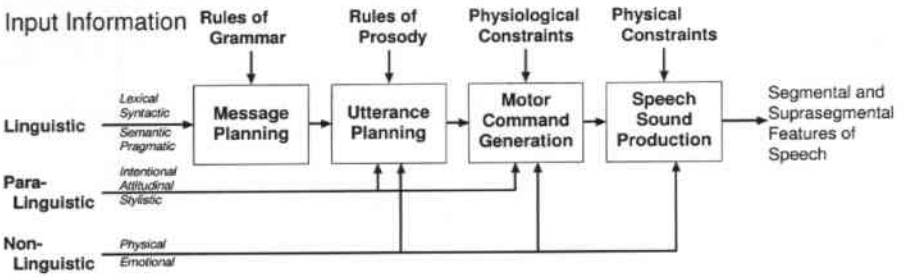


Figure 2.1: Processes by which various types of information are manifested in the segmental and suprasegmental features of speech [8].

The individual characteristics of speech are generated in the process of speech sound production. These segmental and suprasegmental features arise from the influence of linguistic, paralinguistic, and nonlinguistic information. This multi–stage process is displayed in 2.1. This explains the difficulty of finding clear and unique correspondence between physically observable characteristics of speech and the underlying prosodic organization of an utterance.

**Linguistic information:** symbolic information that is represented by a set of discrete symbols and rules for their combination i.e. it can be represented explicitly by written language, or can be easily and uniquely inferred from the context.

**Paralinguistic information:** information added to modify the linguistic information. A written sentence can be uttered in various ways to express different intentions, attitudes, and speaking styles which are under conscious control of the speaker.

**Nonlinguistic information:** physical and emotional factors, like gender, age, happiness, crying, ... which cannot be directly controlled by the speaker. These factors are not directly related to (para-) linguistic contents, but influence the speech anyway.

It is easy to see, how speech can modify and clarify the meaning of a written sentence. A word or just a part of a word in an utterance is emphasized, to accentuate the main information. This stress is sometimes termed as *constructive stress* [15].

As an example, the sentence "I only need a seat on the Shinkansen!" could have three different meanings, depending on the stressed word in the sentence.

"I only need *a seat* on the Shinkansen!"
"I only need a seat on the *Shinkansen*!"
"I only need *a seat on the Shinkansen*!"

The meaning depends on the position and length of stress in this utterance. The speaker adds this kind of paralinguistic information to the sentence to emphasizing the key information. The first utterance shows the listener that the speaker already got a ticket for the Shinkansen, but does not have a seat reservation. The second one means that he has got a ticket for another train, not the Shinkansen. The last one says that the speaker has no ticket yet, but will buy one soon.

In email or other written texts, writers do not have the possibility to assign the paralinguistic and nonlinguistic information to the text. To avoid misunderstandings about the emotional state, so-called *Emoticons* can be inserted to the text in casual communication. `:-)` or `:-(` are two of those *Emoticons* to display emotion. To emphasize a words, capital letters are used.

"I will already come on SUNDAY morning :-)"
clearly differs from
"I will already come on Sunday MORNING :-("

The first one means, that the writer already arrives on Sunday and not on Monday which makes him happy. The latter one expresses that the person does not arrive in the evening, but in the morning. The emoticon clarifies the unhappiness.

In spontaneous speech, capital letters or emoticons do not exist. Instead humans use duration, loudness, and pitch to express such additional information in the utterances. These prosodic features are explained in the following sections.

### 2.1.2 Duration

The duration will be abbreviated as *dur*. Section 3.4 explains that duration is a significant feature to interpret the speech act of a backchannel.

### 2.1.3 Intensity, Power

Surely the intensity resp. the loudness of an utterance is also an essential feature of prosody. In German and English the intensity often marks or emphasizes the central information of a sentence. Without this information, spontaneous speech would often be ambiguous and easily be misunderstood. Since the loudness is measured by the intensity of the energy, this feature will be termed as *pwr*(power).
The power is distributed on the whole backchannel, so it has to be regarded relative to the whole utterance. Therefore, following features are extracted:

**pmean:** the mean power during the backchannel

**pmin:** the minimum power in the backchannel

**pmax:** the maximum power in the backchannel

**ppos:** the position of the maximum power relative to the duration of the backchannel

### 2.1.4 Fundamental Frequency (F0) or Pitch

At the bottom of the humans' vocal tract are the vocal cords, or glottis. For unvoiced speech, the glottis remains open, for voiced speech it opens and closes periodically. The frequency of the opening is called the fundamental frequency.
This frequency, also termed as pitch, can be calculated from the spectrum [2] and its contour over the utterance reveals several information. E.g. in Mandarin Chinese, the F0 carries phonetic/lexical information, and in English or German, the pitch specifies a question by a final fall-rise pattern [15].
This feature plays an important role in the research summed in 3.4 to appoint the meaning of backchannels.

The following features F0 will be extracted from the backchannels for this project:

**fmean:** the mean pitch during the backchannel

**fmin:** the minimum pitch in the backchannel

**fmax:** the maximum pitch in the backchannel

**fpos:** the position of the maximum pitch relative to the duration of the backchannel

**fvcd:** the occurrence of voiced pitch relative to the duration of the backchannel

**fgrad:** the gradient from the position of the minimum and the position of the maximum of the pitch, relative to the duration

### 2.1.5 Glottal Characteristics

Physiological voice characteristics can provide further information for the meaning of backchannels. Helen Hanson's paper[9] describes the interpretation and extraction of glottal characteristics directly from the waveform without the need of special recording equipment.
The following glottal features are extracted from the speech signal at the position with the highest energy:

**adduction quotient (H1-H2):** the difference between the first and the second harmonic indicates a vowel. The value changes when the open quotient rises. Researchers use *H1-H2* as an indication of opening quotient or adduction quotient.

**spectral tilt (H1-A3):** the amplitude of the third formant relative to the first harmonic *H1-A3* is an evidence for spectral tilt and displays the abruptness of the cut off of the airflow.

## 2.2 Clustering

There are several ways to create a classifier. This approach will use unsupervised clustering to get groups of similar vectors. These clusters can be interpreted as classes.
Figure 2.2 gives a short overview of the usability of clustering. A problem

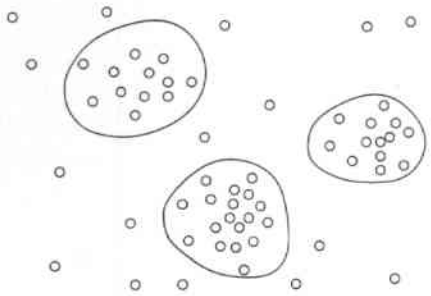

Figure 2.2: Unsupervised clustering in 2D feature space

in this approach is to define the cluster boundaries. A clear solution is often difficult to find.

### 2.2.1 Clustering Method

There are various ways to cluster vector data. The three major methods are *Bottom Up*, *Top Down*, and *k-Means*. This approach also combines these methods to improve the quality of the clusters.

**Bottom Up:** This method starts with as many clusters as vectors in the set. Each vector belongs to its own cluster. The two nearest clusters are merged. This step continues until the number of clusters is reduced to a specific value. The distance of the clusters is declared separately.

**Top Down:** In this method all vectors are initially gathered in one big cluster. Then this cluster is split into two new clusters. Any vectors in this cluster are assigned to one of two new clusters which are represented by two vectors from this cluster. These two vectors can be the two furthest neighbors in the old cluster. Since the calculation of the furthest neighbors costs time, an improved algorithm termed as *fast split*

is established in this project. In latter one, the furthest vector to the average center is calculated. This vector is the first representative of the new cluster, the second one is the furthest neighbor to this vector.

**k–Means:** The *k–Means* algorithm classifies all data to a given number of clusters. The representatives of the clusters are recalculated and the clusters are emptied. Then the data is classified again. With each step, the shape of the clusters become clearer. The algorithm stops after a specific number of steps or a special criteria. The quality of the cluster strongly depends on the initial clusters. In this approach the initial clusters are represented by arbitrarily chosen vectors from the training data set. Three different initial vectors are tested in this project.

**Split & Merge:** This method is a combination of the *Bottom Up* and *Top Down* approach. The widest cluster is split into two new clusters until a certain number of clusters is produced. Then the nearest clusters are merged again. The number to clusters to produce changes with every single step.

**Bottom Up + k–Means:** In this method the clusters will be produced with the *Bottom Up* method explained above. But during the clustering process after a frequently number of merging steps, the *k–Means* algorithm based on the already existing clusters runs. Then the *Bottom Up* continues.

**Split & Merge + k–Means:** This is an extended version of *Split & Merge* algorithm. Every time the algorithm swaps from merge mode to split mode and from split mode to merge mode the *k–Means* algorithm runs to reshape the produced clusters.

### 2.2.2 Cluster Distance

During the clustering, the distance between two clusters has to be calculated. Four types of distance measures are established to get the distance between two clusters. These measurements also depend on the vector distance, which is explained in the section 4.2.3.

**Center Distance:** The average center vector of the cluster is calculated. This center is an artificial vector and does not refer to an actual audio signal. This kind of cluster distance measures the distance between the cluster centroids.

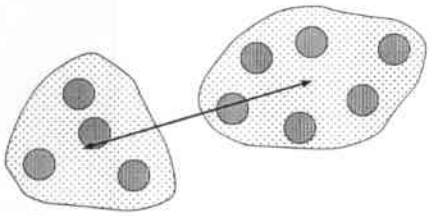

Figure 2.3: Schematic illustration of centroid distance measurement between two clusters.

**Representative Distance:** Different to the center is the representative data vector of the cluster. After the calculation of the cluster's centroid, the nearest vector to the centroid becomes the representative vector. Compared to the centroid distance, the representative distance is less accurate, especially in clusters with a low number of vectors, but its advantage is the high acceleration of the clustering, because a previous calculation of a distance matrix is possible. Additional the representative refers to an audio signal which allows to get an insight to the clusters.

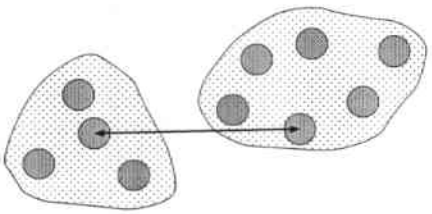

Figure 2.4: Schematic illustration of representative token distance measurement between two clusters.

**Average Vector Distance:** The distance between *all* vectors pairs of both clusters is calculated. The mean value of all this measurement is considered as the distance between two clusters in this method.

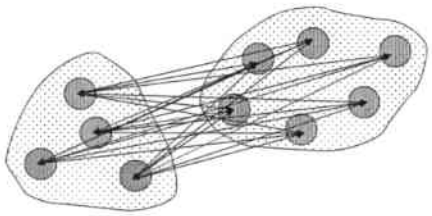

Figure 2.5: Schematic illustration of average token distance measurement between two clusters.

**Furthest Neighbor:** This method measures the distance between the furthest vectors from both clusters, which is the maximum distance of *all* vector pairs between both clusters.

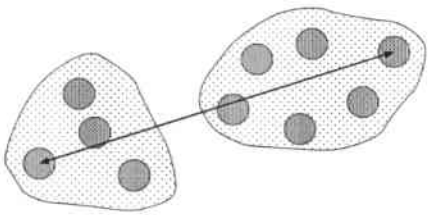

Figure 2.6: Schematic illustration of furthest neighbor distance measurement between two clusters.

These measurements identify a pair of clusters for merging, so it plays a major role during a *Bottom Up* related clustering.
The *nearest neighbor* distance is discarded in this approach, because the earlier project [14] found out, that this method is insufficient for this approach.

## 2.3 Classifier

Clusters can be interpreted as classes. Based on these clusters a classifier tries to find out, to which class a given vectors belongs and returns the according speech act.
To determ the cluster, each cluster has a centroid. The distance between the given vector and each cluster centroid is calculated and the vector is

assigned to the nearest cluster. This method is termed as *nearest centroid* or *Voronoi regions*.
Fig 2.7 shows 2–dimensional Voronoi regions. The boundaries of each region

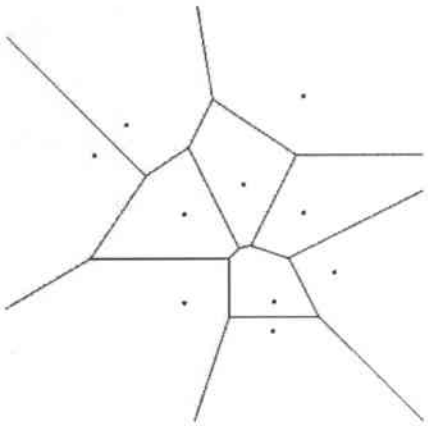

Figure 2.7: Schematic Illustration of Voronoi regions

create a convex cluster.

# Chapter 3

# RELATED WORK

In this chapter, some relevant papers are summarized to give an impression of the current state of the art.

In recent research in LVCSR (large vocabulary continuous speech recognition) filled pauses in spontaneous speech are, if regarded, considered as noise and annoyance for speech recognizers [11]. They were partly detected and discarded. Other research groups consider filled pauses not as arbitrary and cognitive burden. They have communicative functions. Research focusing on prosody uses them to detect utterance boundaries and speech repair, which improves spontaneous speech recognition. Filled pauses even adapt to the situation of the speaker or to the listener. Prosodic features can also be used to detect more information like the emotional state of the speaker.

## 3.1 Die Ikonizität der Pause — The Icony of the Pause

The paper of Kerstin Fischer [7] points to the different functions of filled pauses. Here is exposed that filled pauses are not arbitrary or just sign of cognitive burden. Rather, these pauses have communicative functions in human–human dialogues. This approach tells about experiments, where people had to describe cartoons and interpret them, whereas latter has a higher level of abstraction. The hypothesis, it is possible to draw conclusions between filled pauses and the speech planning design or planning decision, could not be proven, but it was shown that with a higher cognitive burden, the amount of pauses increases. The frequency also seems to be dependent on the length of the utterance.
In human–human conversation, pauses, especially filled pauses, appear more often than in a similar human–machine conversation. In human–human dialogs the cognitive burden was even less than in the experiments before, but more pauses appeared.
This alludes to a communicative function of pauses. Analyses of spontaneous speech demonstrated that filled pauses often indicate utterance boundaries or repair. They often appear after the first word. But additionally filled pauses also act as marking of the central information of the utterance, operate as instrument of self presentation or show dislike of the speaker.
This paper [7] denotes pauses in speech as iconic i.e. that a relation between the speech and the meaning in a structure exists. So it is not arbitrary.

## 3.2 Detection of Phrase Boundaries and Accents

Kießling explains in his paper [3] different prosodic features, which are also relevant for the classification of filled pauses.
Here the researchers analyze the database, which was recorded for the VERBMOBIL project in Germany. It consists of 339 minutes of spontaneous speech of 56 female and 81 male speakers. In this paper is explained, how to detect accents and phrase boundaries in spontaneous speech.
The following features are computed for the detection and classification of each syllable.

**PAUSE:** The pause following the syllable

**DUR:** The duration of the syllable nucleus and the relative duration of the whole syllable

**RATE:** The average speaking rate of the whole utterance

**F0:** Different features extracted from the F0-contour, like the regression coefficients *(F0reg)*, the onset, offset, minimum and maximum F0 *(F0val)* and their positions *(F0pos)* on the time axis relative to the center of the syllable

**INTENS:** The maximum intensity and its positions relative to the center

**FLAGS:** Flags for indicating word finals or lexical accents

## 3.3 Filled Pauses in Spontaneous Speech

The author Batliner analyzes in his paper [1] filled pauses in a database, which was recorded for the VERBMOBIL project in Germany. These German dialogs contain 339 minutes of spontaneous speech of 56 female and 81 male speakers.
They detected several kinds of filled pauses (FP) and analyzed especially two types of them, the filled pause as a hesitation (FPH) and as repair (FPR). In about 42% of the cases FP are adjacent to breathing, silent pauses or word lengthening. Thus, hesitations are almost always signaled either by FPH or by lengthening but not by both. In average, FPs amount to 2% of the vocabulary i.e. a FP occurs in almost every second turn.

## 3.4 Analysis of Para-linguistic and Non-linguistic Information based on Acoustic Parameters of Aizuchi

The master thesis of Umeno [13] analyzes para-linguistic and extra-linguistic information in spontaneous speech. The research is limited to the single filled pause, which is transcribed as "un". This is equivalent to the English filled pause "mhm".
In this approach, about 5 hours extracted from 150 hours spontaneous everyday speech are taken, although the amount of utterances is limited to 2500

samples of "un".
PCA analysis reveals that speaker attitude and speaker characteristics are perceived separately.
Analysis of acoustic parameters of the speaking style features confirmed that the speaker adapts her speaking style according to listener i.e. if the counterpart is a relative, a friend or a superior.
The utterances are clustered in three classes ("Yes", "No", "Filler"). 20 listeners judged the appropriateness of the three types of utterance to the various discourse context. F0 contour and duration of the signal are extracted for the clustering.
The evaluation exposes that these three cases occurred frequently:

- short duration, high F0: shows affirmation or affirmative listening.
- short duration, low F0: shows affirmation
- long duration, low F0: shows denial

These results target a better speech synthesis for ESP Project.

This research will deal with the same database and is similar to the latter part of Umeno's research.

## 3.5 Detecting Emotion in Speech

The approach of Polzin [12] demonstrates that human language can be explored by means of acoustic and prosodic features to detect emotion in human speech. A suprasegmental HMM (SPHMM) results from the combination and integration of prosodic information and acoustic information within a HMM architecture.
Five drama students were asked to speak 50 sentences each, pronounced with 5 different emotions *(happy, sad, angry, afraid, neutral)*. An emotion-dependent acoustic and suprasegmental model was trained. Evaluation showed, that both human and machine have a similar emotion recognition accuracy of about 70%.

## 3.6 Deviations

Polzin's work deals with data recorded from drama students. This is not really spontaneous and natural speech. Umeno's data, the same data which is used in this project, consists of recorded every-day-speech, and is therefore more natural than the speech of drama students. However this research will analyze the data from the aspect of speech recognition and semantic analysis. Filled pauses like "un" have more meanings than only "Yes" and "No". In spontaneous dialogs they play an important role for the fluency of the turn. E.g. they can denote the speaker to continue his turn or demonstrate the attention of the listener. This is why they are so important for interpretation of a dialog.

# Chapter 4

# CLASSIFICATION METHODS

In this chapter, the experimental setup is explained, how the data will be clustered and how the classifier is created.

## 4.1 Experimental Setup

In the experiments the tokens will be represented by prosodic feature vectors. These vectors will be clustered and a class will be assigned to these clusters.

### 4.1.1 Prosodic Feature Vectors

First of all, the audio signals (tokens) need to be prepared for the experiments. The tokens are represented by a single prosodic feature vector. The feature set is explained in section 2.1.1 and consists consists of *dur, fmean, fmax, fmin, fpos, fvcd, fgrad, pmean, pmax, pmin, ppos, H1-H2,* and *H1-A3*.
During the clustering these 13-dimensional vectors are used for distance measure and centroid calculation. To get normalized values each coefficient is normalized to a value between 0 and 1. The floor and ceiling values are

taken from the observation of the data and are rounded values of the maximum and minimum of the feature vector coefficients. Few outliers (less than 1%) are ignored for these values and the coefficients are clipped because a higher ceiling or a lower floor value would reduce the feature's significance. Table 4.1 illustrates the floor and ceiling values if the single features.

| feature | floor value | ceiling value |
|---|---|---|
| dur | 0 | 3 |
| fmean | 0 | 400 |
| fmax | 0 | 400 |
| fmin | 0 | 400 |
| fpos | 0 | 1 |
| fvcd | 0 | 1 |
| fgrad | -15 | 15 |
| pmean | 0 | 3000 |
| pmax | 0 | 3000 |
| pmin | 0 | 3000 |
| ppos | 0 | 1 |
| H1-H2 | -50 | 50 |
| H1-A3 | -50 | 80 |

Table 4.1: Vector bounds for feature normalization.

### 4.1.2 Data Sets

For the experiments the data is divided into three subsets. The first set consists of the 2167 unlabeled backchannels and will be the *training set*. The feature vectors extracted from these tokens are used for the clustering.
The labeled data is segmented into two subsets, the *development set* and the *test set*. The tokens are chosen in order to have the same quantity of speech acts over both sets. The frequency of the labels in the sets is illustrated in Table 4.2.

The *development set* is used to assign labels to the clusters and create a classifier which allows the interpretation of feature vectors. Furthermore quality issues of the clustering method can be considered by analyzing the separability of the classes.

| number | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| dev | 189 | 23 | 10 | 7 | 2 | 1 | 7 | 3 |
| test | 188 | 23 | 9 | 7 | 3 | 0 | 8 | 2 |

Table 4.2: Frequency of the label numbers in the development set (dev) and the test set (test).

In usual terminology, the purpose of the "development set" is to fine-tune the classifier. In this research it is also considered as a labeled training set. The vectors in the *test set* are used to calculate the recognition rate and evaluate the classifier. The functionality of the classifier is explained in section 4.3.

### 4.1.3 Configuration

Figure 4.1 gives a short overview of the experimental configuration.

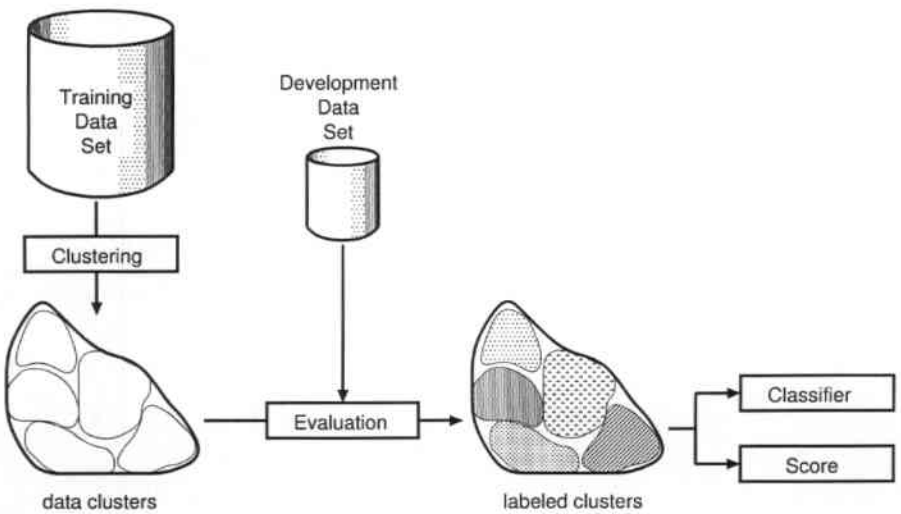


Figure 4.1: The configuration of the clustering algorithm.

The *training set* consists of the feature vectors of the unlabeled tokens. As illustrated in Figure 4.1 the training set is clustered into several clusters.

This is an unsupervised automatic clustering.
Each cluster has a centroid. Using the *nearest centroid* method the vectors in the *development set* are classified to the resulting clusters. Depending on the distribution of the vectors in the clusters labels are assigned to the clusters.
With the knowledge of the centroids and the assigned label, a classifier is created. Due to the ambiguous labels in some clusters, high inhomogeneous clusters are subclustered and labels are reassigned to the subclusters.

## 4.2 Parameters

Since there are several ways for clustering the following parameters define the experimental setup. Different setups will be tested in during the experiments.

### 4.2.1 Clustering Method

In chapter 2.2.1 the different methods for clustering are explained. These methods are *Bottom Up, Top Down, k–Means, Split & Merge, Bottom Up + k–Mean*, and *Split & Merge + k–Means*.

### 4.2.2 Cluster Distance

The different cluster distance measures are explained in section 2.2.2. In the experiments a distinction is drawn between the following methods: *center distance, representative distance, average vector distance*, and *furthest neighbor*.
This distance measure is necessary to find the closest pair of clusters to merge.

### 4.2.3 Vector Distance

To classify vectors or to check the distance between clusters the distance measurement between two vectors plays an important role.
This approach uses the *Euclidean distance* between two vectors. To emphasize certain features a weighting is established. The weighting is a vector

with 13 coefficients. Each one represents a factor which is multiplied to the feature value during the calculation of the metric of the vector.
This method corresponds to stretching or compressing of the feature space.

$$\sqrt{\sum_{f \in features}(x_{1_f} - x_{2_f})^2 * weight_f}$$

Each weighting is assigned a number. After the visualization and an examination of the data, the following five weightings were defined. *Weighting 0* means no alteration of the feature space, while *weightings 1 to 4* influence certain features. *Weightings 5, 6, and 7* are established later after the experiences with the first five weightings. Chapter 5.2 contains a detailed explanation how these weightings are calculated.

| weighting | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| fmean | 1.0 | 1.0 | 1.0 | 2.0 | 2.0 | 0.71 | 1.0 | 1.0 |
| fmax | 1.0 | 0.0 | 1.0 | 1.0 | 1.0 | 0.59 | 0.0 | 1.0 |
| fmin | 1.0 | 0.0 | 1.0 | 1.0 | 0.0 | 0.47 | 0.0 | 0.0 |
| fpos | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.71 | 1.0 | 0.0 |
| fvcd | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.44 | 0.0 | 0.0 |
| fgrad | 1.0 | 1.0 | 1.0 | 2.0 | 2.0 | 0.62 | 1.0 | 1.0 |
| pmean | 1.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.47 | 0.0 | 0.0 |
| pmax | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.71 | 1.0 | 1.0 |
| pmin | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.50 | 0.0 | 0.0 |
| ppos | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.27 | 0.0 | 0.0 |
| dur | 1.0 | 1.0 | 1.0 | 2.0 | 2.0 | 1.00 | 1.0 | 1.0 |
| H1-H2 | 1.0 | 0.0 | 1.0 | 2.0 | 0.5 | 0.50 | 0.0 | 1.0 |
| H1-A3 | 1.0 | 0.0 | 1.0 | 2.0 | 0.5 | 0.23 | 0.0 | 0.0 |

Table 4.3: Different weightings for vector distances.

### 4.2.4 Number of Clusters

The number of clusters in the classifier can be critical in the clustering process. It influences the runtime of the algorithms and the quality of the classifier. In this approach this parameter is not explicitly regarded. In an early approach [14] the number of 15 clusters is considered as adequate. This

parameter will be set to 20 or 40 during the experiments, to compare the quality of the classifier depending of the size of the clusters.

## 4.3 Classification

Each cluster represents a specific group of vectors with specific characteristics. The goal of the clustering process is to create classes as accurate as possible to represent a certain speech act.
To find out the speech act of a given token, the feature vector is extracted from the audio signal and classified to a cluster, which stands for a specific speech act.
Each cluster can be represented by two vectors. One vector is the average cluster center (centroid), an artificial vector created with no audio signal. The other vector is a representative vector from the cluster, which is the nearest vector to the centroid.
To *classify* a vector to a cluster, the distance to each cluster representative (or centroid) is calculated and the vector is assigned to the cluster according to the minimum distance. This value depends on the feature weighting described in 4.2.3. This approach uses exclusively the cluster representative vector.
The clusters are created on the *training set* and this is why they contain no knowledge about their speech act. The *classifier* assigns a feature vector to one of these clusters *and* to an according speech act. To get latter information the vectors in the *development set* are assigned to the clusters and after that, a meaning is interpreted from the frequency of the labels in the clusters. It often occurs that the speech act is not easy to decide, because the cluster contains vectors with differing ambiguous labels. The speech act is decided by the highest number of vectors of the class relative to the absolute vectors in the class.

Table 4.4 shows a short excerpt of the classification table. This experiment is based on a *Bottom up + k-Means* algorithm with *furthest neighbor* distance and using *weighting 1*. The final number of clusters is 20.
The problem is the assignment of a speech act to a cluster. As already mentioned above, the speech act derives from the highest number of vectors relative to the absolute vectors in the class. According to Table 4.2 the absolute number of vectors with label 1 in the *development et* is 189. Cluster

| label | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| Cluster 1 | 80 | 7 | 2 | 0 | 0 | 0 | 0 | 1 |
| | 42.3% | 30.4% | 20.0% | 0.0% | 0.0% | 0.0% | 0.0% | 33.3% |
| Cluster 2 | 43 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| | 22.8% | 0.0% | 10.0% | 14.0% | 0.0% | 0.0% | 0.0% | 33.3% |
| Cluster 3 | 11 | 1 | 0 | 0 | 0 | 0 | 5 | 1 |
| | 5.8% | 4.3% | 0.0% | 0.0% | 0.0% | 0.0% | 71.4% | 33.3% |

Table 4.4: Short excerpt of classification table. Upper row is the absolute number of vectors classified to that cluster, second row displays the percentage share of vectors labeled with this number.

1 is assigned to label 1 (listen) because 42.3% of the vectors with label 1 are classified to this cluster, which is the percentage share maximum. Cluster 2 shows a complete different situation. Although 43 vectors of label 1 are in this cluster, it would be interpreted as label 8, because 43 vectors match "only" 22.8% while 1 vector of label 8 matches 33.3%. So 45 vectors in cluster 2 will be misclassified and only one correctly. The situation in cluster 3 is easier. 5 vectors will be correctly classified as label 7 while 13 vectors are misclassified.
Merging the classes changes the percentage share of the vectors in the classes. Class 7+8 now contains 10 vectors. The one vector with label 8 in Cluster 2 now matches 10.0% of class 7+8, while 43 vectors with label 1 now match 20.3% of class 1+2. So after the class merging, cluster 2 will be interpreted as class 1+2.

If after the class merging, the cluster is still very non-uniform, it will be subclustered into smaller clusters and a new meaning is assigned to the smaller subclusters. The method for subclustering is *k–Means* and it clusters the part of the data into $\log n$ clusters, while $n$ is the number of vectors in the original cluster.
A cluster is subclustered, if two or more classes match 20.0% or higher.

The *classifier* saves the representatives of the clusters and the speech act which is assigned to the cluster. Now any feature vector can be classified and interpreted.
If a cluster is subclustered, the representative of this cluster gets a special label. This label "points" to the representatives of the subclusters.

If a vector is classified to a subclustered cluster, the classifier reclassifies this vector by the use of the subcluster representatives.
Since the classifier is a nearest neighbor classification, the inclusion of the subcluster representatives in the first level would resize the original and the neighboring clusters. This is why the second level is established to stabilize the original classes.

## 4.4 Score and Evaluation

As the example in Table 4.4 shows, it is critical to assign the speech act to a cluster. Usually the *average class recognition rate* evaluates the clusters.

**Average Class Recognition Rate (ACRR):** The average class recognition rate *(ACRR)* is the mean percentage share of how many data is classified to the correct class, relative to the number of data in this class.
In general it is the mean value of the diagonal in a confusion matrix.

**Token Misclassification (TMCL):** The token misclassification *(TMCL)* value corresponds to the percentage share of how many data is misclassified relative to absolute number of data.

Since this is an unsupervised approach some automatic criterion has to be established how to assign the classes to the clusters. This criterion is to achieve an ACRR as high as possible. This is why in the example in section 4.3 the cluster 2 is assigned to class 8 and not to class 1. This decision leads to the maxima maximization of ACRR.
Also observable in this example is, that 45 vectors are classified wrong while only one is classified correctly. If class 1 would be ignored completely and a (theoretical) perfect separation of all other classes is achieved, the ACRR would have a value of 87.5%, *but* 78.2% of all data would be classified incorrectly. Classify all vector as class 1 (like state of the art speech recognizers do, interpret as backchannel) would achieve a TMCL of only 21.8%, but an ACRR of 12.5%.

So *both* values are very important to gain an objective evaluation of the clusters. Therefore a *score* is established which is calculated from the ACRR

and TMCL. In the final classifier, the classes 1 and 2, 4 and 5, and 7 and 8 will be merged. The classifier is recalculated because the proportion of the classes differ after the merging. In the given example cluster 2 would not be classified as class 8 (or 7+8 after the merge) anymore, because only 1 vector of this new class corresponds to 10.0% while class 1+2 now includes 212 vectors and the 43 found vectors represent 20.3% of this class. In this case cluster 2 will classify a vector as class 1+2.

$$score_{dev} = (ACRR_{nomerge} - TMCL_{nomerge}) + (ACRR_{merged} - TMCL_{merged}) \quad (4.1)$$

$$score_{test} = (ACRR_{merged} - TMCL_{merged}) \quad (4.2)$$

$score_{dev}$: This score is calculated during the evaluation phase of the experiments. The equation is given in 4.1. It is a close evaluation, independent from the *test set* and allows a rough estimate of the cluster quality.
This score includes information of the quality of the clusters *before and after* merging the classes so it is in the range from -200 to +200. This "double" score shall prevent to categorize clusters as good, although they are not before the merging. The score can only be high, if both results are good. A more accurate estimate can be achieved this way.

$score_{test}$: Since the ACRR does not give any information about the number of misclassification the results on the *test set* will be represented by this score in a range from -100 to +100 (4.2). The data is tested with the final classifier after the merges.

# Chapter 5

# EXPERIMENTS & RESULTS

## 5.1 Data

### 5.1.1 Speech Data

The data for the experiment is taken from the JST/CREST ESP (Japan Science & Technology Agency / Core Research for Evolutional Science and Technology Expressive Speech Processing) project [6] and contains about 150 hours of every-day speech from one speaker. First examinations of the data in Umeno's master thesis [13] observed nonlinguistic and paralinguistic information in a subset of 5 hours of backchannels.
Most current emotional speech databases are artificially created and therefore fail to adequately represent spontaneous speech of a common person's everyday conversation. These databases are limited in their ability to illustrate supra-linguistic speech variation. The paper about the recording of the data [4] explains an alternative method for database design.

> A bottom-up alternative to database design would require a different type of approach, based not on the need of the researchers, but on the habits of the user community, as defined by their everyday language use.

In the five-year ESP project, the goal is to collect natural everyday speech. With this corpus it is possible to design speech technology applications which

can handle changes in speaking style and voice quality. These changes indicate the intentions in each utterance. Because of the wide range of everyday situations, the collection targets ordinary people, which express various attitudes and emotions.
The major disadvantage of this kind of data is the noise in the background. Typical interferences are passing vehicles or a screaming baby. This problem takes a high influence to the recording quality. Modern speech recognizers still have problems to deal with this kind of noise.
Recording devices (in the form of a microphone, a recording engineer, etc.) can influence the speaking style of the proband. This problem is called *Observer's Paradox*. The person will switch into a public or self-conscious mode of behavior, because of the awareness of having a microphone in front of him. Therefore *observer-free recording* is necessary. Such recording shall record from morning to night, including all situations of interaction with other people like family members, friends, traders, strangers, colleagues, etc.
The recording apparatus has to be user-friendly and shall impose as few constraints on the speaker as possible. Lightness and wearability have to be considered.
The ESP approach is to have volunteers wear light head-mounted studio-quality microphones, suspended from the ears and largely hidden by the hair. After some time, the speaker gets accustomed to the apparatus and adapts his behavior, so no touch on the microphone or other equipment-related noise will occur.
The speakers use different recording devices such as radio transmitters, a direct connection to a DAT recorder, or a portable Minidisc recorder. The size and weight of DAT recorders will exclude the customization to the device, so the volunteer will react unnaturally towards other people. This makes the Minidisc recorder the most "wearable" recording device for application in the field, i.e. on the street, in shops, at home etc. Although the Minidisc recorder applies lossy compression on the signal, Nick Campbell [5] suggests that both recording media (DAT and MD) can be considered equivalent for the purpose of prosodic analysis of human speech.

The word "utterance" in usual terminology means an audio signal of a complete recorded sentence. Although for this research an "un" is a complete utterance, the term "token" will be used whenever referring to a segment of a speech signal containing a single backchannel.

### 5.1.2 Labels

2649 tokens of the manually segmented backchannel "un" are available, whereof 482 are labeled by one human expert. The data is labeled with a number between 1 and 8, while each number is an synonym for a specific speech act. The number, the according speech act, and the frequency of the labels is illustrated in Table 5.1. The distribution of the speech acts is non-uniform.
The remaining 2167 unlabeled backchannels are used for the unsupervised data-driven clustering.

| number | speech act | quantity |
|---|---|---|
| 1 | listen | 377 |
| 2 | understand | 46 |
| 3 | interest | 19 |
| 4 | affirm | 14 |
| 5 | affirm overall | 5 |
| 6 | callback | 1 |
| 7 | disagree | 15 |
| 8 | emotion | 5 |

Table 5.1: Label number and according speech act.

### 5.1.3 Merging Classes

A large labeled data set is needed to create an accurate classifier. A small amount of vectors is not enough to represent a speech act class. Especially the classes 5, 6, and 8 contain very few tokens. An additional issue is, that the data is divided into the development and test set, which downsize the classes to train the classifier.
The early approach [14] analyzed the behavior of the classes in the feature space and found out, that classes 1 and 2, classes 4 and 5, and classes 7 and 8 hardly differentiate each other. According to the speech act, this absolutely makes sense. Visual observation of the feature vectors and listening to tokens of the classes proved this assumption. This is why these pairs of classes are merged and retermed.

| number | speech act | quantity |
|---|---|---|
| 1+2 | listen | 423 |
| 3 | interest | 19 |
| 4+5 | agree | 19 |
| 6 | callback | 1 |
| 7+8 | disagree | 20 |

Table 5.2: Merged classes and according speech act.

Table 5.2 show the distribution of the classes and tokens after the merging. The quantity of the tokens in the smaller classes is increased.
Class 6 is ignored in the test. It is impossible to train and test a classifier with one token.
Figure 5.1 illustrates the training and the development set in a 2 dimensional view. Duration and pitch is alluded to Umeno's observations [13] and proves his theory, that positive backchannels have a short duration and a high pitch, while negative backchannels have a long duration and low pitch. The development set covers a large part of the training set, although the training set is more than 8 times larger. Class 1+2 covers almost the complete represented space, while class 3 is scattered. Class 7+8 distinguish from other classes except 1+2, while class 4+5 is mixed with classes 1+2, 3, and 6.
The illustration also give information, that the classes are separable, but 2 dimensions as in this simplified figure will not be enough.

### 5.1.4 Additional "Honma" Data

To verify the portability of the results from the "un" experiments to other backchannel, the ESP project kindly provided additional data of the backchannel "honma". This can roughly be translated as "oh really?". This set consists of only 393 tokens, there is no additional unlabeled training set. 315 of the tokens are labeled to one out of 13 speech acts. These speech acts are *impressed, ok, backchannel, question, satisfied, unhappy, understanding, laugh, disgusted, doubtful, disappointed, surprised,* and *happy*.
The tokens were labeled by at least seven (up to nine) labelers, who assigned the token to the according group. If there is no recognizable majority in the

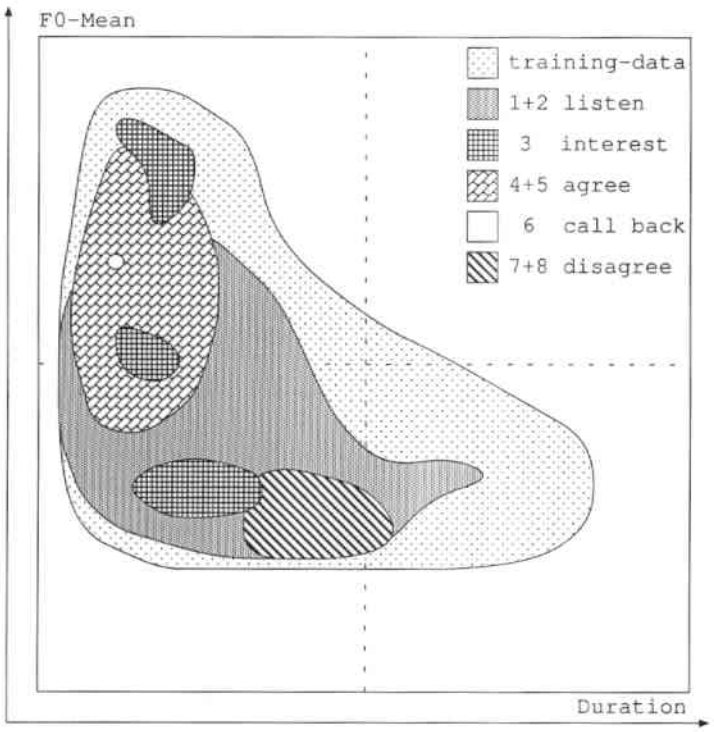


Figure 5.1: Visualization of training and development set.

experts' opinion, this token is labeled as *mixed*.
The distribution of the labels in the "honma" data set is shown in 5.3.

## 5.2 Optimal Feature Weighting

In section 4.2.3 is mentioned, that a weighting of the single features during the token distance measure is possible.
To find an optimal feature weighting, a clustering algorithm has to test all combinations of features. In this test, all weighting factors are set to 0 or 1. Since the feature vector contains 13 coefficients there are $2^{13} = 8192$ possible combinations to test. This is why the clustering algorithm has to be fast to gain results in an appropriate time. It is decided, that the feature space has to be at least 3 dimensional. This constraint reduces a little the number of

| speech act | quantity |
|---|---|
| impressed | 24 |
| ok | 14 |
| backchannel | 45 |
| question | 19 |
| satisfied | 4 |
| unhappy | 8 |
| understanding | 24 |
| laugh | 7 |
| disgusted | 5 |
| doubtful | 24 |
| disappointed | 6 |
| surprised | 18 |
| happy | 32 |
| mixed | 85 |

Table 5.3: Labels and according frequency in "honma" data set.

combinations to 8100.

The combination of *Split & Merge* and *k-Means* algorithms has established as fast and accurate clustering algorithm. The experiment table in the Appendix lists 73 experimental setups. The experiments #38, #39, #40, #41 and #52, which use this algorithm, achieve a mean development score of 70.2. The setups just vary in the feature weighting. All feature weightings used until that moment are tested in these experiments and can be compared.
Weighting 5, 6, and 7 are derived from the following experiments.

The training set is reduced to 1500 randomly chosen vectors from the original training set. The experiments are repeated with the reduced set so check, if the set is still representative. The clustering with the new set achieve a mean score of 52.9. As expected, the results are worse, but still high.
The set is clustered 8100 times with different feature weighting combinations. All clusters are evaluates as explained in section 4.4 and the features can be compared now.

The histogram 5.2 shows the frequency of the features in percent dependent to the score the experiment achieved. The first bar display the frequency of the features in experiments, which achieve a development score *lower than -50.0*. The second and third illustrate the feature frequency in the experiments with a score *higher than 20.0 resp. 90.0*.
Features with a rising frequency like *pwrMax*, *f0mean*, and especially *dur* seem to be important for a "good" clustering. In the experiments with an insufficient score, these features are little participated.
Contrary term for features with decreasing frequency such as *H1-H2*, *f0vcd*, and *H1-A3*. In non reasonable experiment results, these features are more participated than in sufficient experiments.
For example, the feature *dur* is used in *all* experiments which have a score over 90.0, but only in 6.6% of the experiments with a score under -50.0. *H1-A3* is found in 68,4% of the experiments with the score lower than -50.0 and only in 23.5% of the cases over 90.0.

Interesting is the relevance of features which have a frequency of about 50.0% like *pwrMean* and *pwrMin*. These features do not influence the quality very much, because it works *with and without* them.
Each factor in *weighting 5* is taken from this histogram and is the frequency of the features in the experiment which achieved a score higher than 90.
*Weighting 6* is the discrete version of weighting 5. All factors are 1.0 or 0.0. Only two combinations were able to achieve a score over 100. These are established in *weighting 7*.

As conclusion can be considered that each feature has its own relevance for the clustering. Some features seem to carry a lot of information to shape adequate clusters. These features are *f0Mean, pwrMax, f0Pos, f0Grad*, and *dur*. Other features seem to deteriorate the quality of the clusters such as *f0vcd, H1-H2*, and *H1-A3*.
The remaining features (*pwrMean, pwrMin, f0Min, pwrPos, f0Max*) are redundant and do not change the quality significantly.

## 5.3 Results

72 different experimental setups have been tested during the experiments. A complete schedule of the experiments can be found in the Appendix. How

to read the table is explained in the discussion chapter 6.1.
Each experiment gave conclusions to improve the performance of the later clustering algorithm. So a way to find an optimal feature weighting was found out during the experiments. This feature weighting further improved the performance of the classifier. Section 5.2 explains the experiments to find the optimal feature weighting.

After all a *Bottom Up* clustering using the *furthest neighbor* cluster distance and feature *weighting 5* performed best. The training set is clustered into 20 clusters.
Table 5.4 shows the confusion matrix of the classifier.

| | classified as | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | listen | | interest | | agree | | disagree | |
| class | n | % | n | % | n | % | n | % |
| listen | 181 | 85.8 | 5 | 2.4 | 9 | 4.3 | 16 | 7.5 |
| interest | 3 | 33.3 | 4 | 44.4 | 2 | 22.2 | 0 | 0.0 |
| agree | 2 | 20.0 | 1 | 10.0 | 7 | 70.0 | 0 | 0.0 |
| disagree | 4 | 40.0 | 0 | 0.0 | 0 | 0.0 | 6 | 60.0 |

Table 5.4: Confusion matrix on test set. Absolute number (n) and percentage of vectors in each class is given.

The test set consists of 240 tokens. The classifier classifies 198 tokens (82,5%) correctly and achieves an class recognition rate of 65.1%.

## 5.4 "Honma" Experiments

The additional "honma" data should run on similar experiments. This data set consists of less data then the "un" data set.
393 token are available, 315 of them are labeled to the speech acts explained in section 5.1.4. The 78 unlabeled tokens come with a insufficient recording quality, this is why each experiment run twice in each case with a different training set. The one training set consists of the development set and the unlabeled data, the other set only consists of the development set. To increase the training set in both cases, the development and testing set is not

segmented in equal size this time. The segmentation of the set is given in Table 5.5.

| speech act | dev | test |
|---|---|---|
| impressed | 16 | 8 |
| ok | 9 | 5 |
| backchannel | 30 | 15 |
| question | 13 | 6 |
| satisfied | 3 | 1 |
| unhappy | 5 | 3 |
| understanding | 16 | 8 |
| laugh | 5 | 2 |
| disgusted | 3 | 2 |
| doubtful | 16 | 8 |
| disappointed | 4 | 2 |
| surprised | 12 | 6 |
| happy | 21 | 11 |
| mixed | 56 | 29 |

Table 5.5: Frequency of the labels in the development set (dev) and the test set (test).

To assign the labels to the clusters, again the development set is used. The test is only for testing.

The data is clustered with the same 72 different experimental setups. The results of the experiments are discussed in the section 5.4.

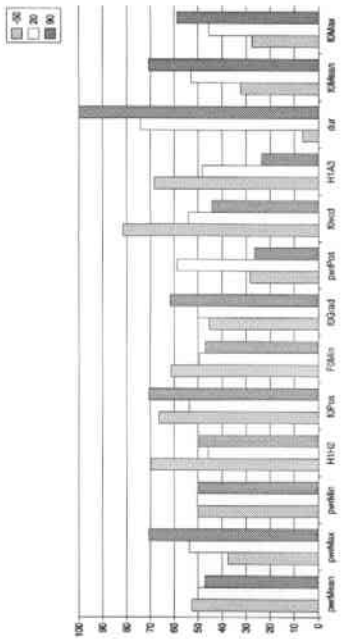


Figure 5.2: Feature frequency in different experiments.

# Chapter 6

# DISCUSSION

This chapter discusses the performance of various experimental setups. Each parameter influences the quality of the clusters.
The ability to transfer this approach to the backchannel "honma" is discussed later in this chapter.

## 6.1 Clustering performance

Table 6.1 shows a short excerpt from of the results of the classifier. These examples are taken from the complete table in Appendix and illustrate the different characteristics of the experimental setups.
The first column (#) shows the number of the experiment, the next (*Method*) illustrates the clustering algorithm, while the column *Distance* gives information about the distance measure between the clusters. *#Wgt* is the number of the used feature weighting and #Clust shows the number of clusters the data is clustered to. The column *Dev score* gives the development score in this experiment. The Averaged Class Recognition Rate (*ACRR*) and the Token Misclassification (*TMCL*) on the test set are shown in the next two columns. The last (*Test score*) is the test score which is the difference of the ACRR and the TMCL. This value is the main indicator for the quality of the experimental setup. See section 4.4 for further details of the scores.

| # | Method | Distance | #Wgt | #Clust | Dev score | ACRR | TMCL | Test score |
|---|---|---|---|---|---|---|---|---|
| 1 | Bottom Up | Center | 0 | 20 | 4.0 | 49.9 | 34.4 | 15.5 |
| 3 | Bottom Up | Furthest | 0 | 20 | 77.4 | 60.9 | 24.9 | 36.0 |
| 4 | Bottom Up | Average | 0 | 20 | -87.3 | 43.2 | 51.5 | -8.3 |
| 5 | Bottom Up | Represent | 0 | 20 | -87.6 | 60.0 | 43.6 | 16.4 |
| 9 | Bottom Up | Furthest | 1 | 20 | 84.7 | 53.2 | 43.6 | 9.6 |
| 15 | Bottom Up | Furthest | 1 | 40 | 95.0 | 53.3 | 52.3 | 1.0 |
| 17 | Top Down | Fast | 0 | 20 | -71.6 | 42.1 | 56.4 | -14.3 |
| 19 | Top Down | Fast | 1 | 20 | -0.6 | 61.2 | 37.3 | 29.9 |
| 20 | Top Down | Furthest | 1 | 20 | 6.0 | 59.3 | 34.9 | 24.4 |
| 34 | k-Means | Random 3 | 2 | 20 | 7.5 | 58.5 | 49.6 | 8.7 |
| 38 | S&M + k-Means | Furthest | 0 | 20 | 74.1 | 61.3 | 50.2 | 11.1 |
| 39 | S&M + k-Means | Furthest | 1 | 20 | 49.9 | 66.0 | 48.6 | 17.4 |
| 41 | S&M + k-Means | Furthest | 3 | 20 | 68.6 | 55.1 | 25.3 | 29.6 |
| 54 | Butt.Up + k-Means | Furthest | 4 | 20 | 52.4 | 52.0 | 29.9 | 22.1 |
| 56 | Bottom Up | Furthest | 5 | 20 | 86.2 | 64.5 | 19.5 | 45.0 |
| 64 | Split & Merge | Furthest | 6 | 20 | 88.7 | 66.9 | 32.4 | 34.5 |
| 73 | S&M + k-Means | Furthest | 7 | 20 | 26.2 | 51.9 | 28.2 | 23.7 |

Table 6.1: Excerpt from the result table of the different experimental setups.

### 6.1.1 Clustering Method

The *Bottom Up* clustering method achieved both the worst (-87.3) and best (95.1) development score during the experiments. Various parameter combinations are tested with this method. Sufficient results are performed with the different weightings and the *furthest neighbor* distance measure. The averaged token, centroid and representative token distance do not lead to a high development score.
The *Bottom Up* clustering method is the slowest one. All clusters have to be compared with each other which is very time consuming in the beginning, when all feature vectors represent a cluster. This algorithm runs in O((n-k)*n*n) while n is the number of tokens and k is the number of clusters.

Using the *Top Down* method for clustering no sufficient performance can be achieved. In most cases the development score is lower than 0.
Two different types of splitting the clusters are tested, but neither the *furthest neighbor* in the cluster nor the *fast split* lead to sufficient results. In two of three cases the furthest neighbor split leads to a better development score than the fast split. The reason for this observation is probably, that furthest neighbor is more accurate than the fast split. The latter one just guesses the two neighbors which are used for splitting. This is why this furthest neighbor splitting is considered as more adequate for this clustering method.

The *Split & Merge* method, which is a combination of both *Bottom Up*

and *Top Down* shows a similar behavior like *Bottom Up*. Depending on the weighting of the features, this method achieves a low score (-55.6) but also a high score (62.5). The *Top Down* experiments show that a furthest neighbor split perform better than the fast split. The furthest neighbor in the *Bottom Up* part of this method is also considered as the most adequate measurement.

The performance of the *k–Means* method depends on the initialization. This algorithm tries to create uniform distributed clusters in feature space. These clusters achieve in most experiments a positive development score, but still not sufficient. The maximum development score which is achieved is 59.0. Three randomly initialized start sets are tested. In two of three experiments the third set (*Rand 3*) leads to the best score.
The main advantage of this method is the runtime, *k–Means* runs quicker than other methods.

The performance of the *k–Means* depends on the initialization. A combination of *Bottom Up* and *k–Means* methods shape these initial clusters by using *Bottom Up*. Also the *k–Means* reshape the clusters during the *Bottom Up* part.
The combination of both methods seems so be an adequate method for clustering the training data. A very high development score (87.4) can be achieved.

In earlier research [14] *Split & Merge + k–Means* turned out to be the best method. In this approach, the feature weighting in the second level of the classifier was adjusted manually to improve performance.
The current approach now does not vary the feature weighting in the second level. The software is improved and automatically chooses the clusters for second level subclustering. The classifier still leads to similar results. Experiment *#61* achieves a development score of 82.7 and a test score of 33.9.
This method is sufficient in both runtime and classifying performance. This is why this algorithm is chosen for the investigation of the optimal feature weighting explained in 5.2.

### 6.1.2 Cluster Distance

Besides the clustering methods, the cluster distance takes high influence to the cluster quality, this is why it is an important parameter. Table 6.1 shows among other things the results of experiments #1, #3, #4, and #5.

Compared to other methods *furthest neighbor* is the only method which achieves high positive development scores, while centroid distance (*center*) perform a development score of about 0. The distance between the representative cluster tokens (*represent*) and the averaged token distance (*average*) lead only to a negative development score.
An explanation for the insufficient results of the averaged token distance needs further investigation, while the bad performance of the representative token distance is probably underlying the inaccuracy of the measure in small clusters.

When a cluster is split like in *Split & Merge* or *Top Down* this parameter also controls the calculation of the two representatives of the new clusters. The experiments (e.g. #19 and #20) show that *furthest neighbor* is also the best method to split a cluster.

### 6.1.3 Feature Weighting

The experiment which is explained in Section 5.2 proved, that some features seem to be more relevant than others. The 8 different feature weightings in Table 4.3 are chosen in the clustering experiments.
Although the weighting improves the classification performance, the usefulness of the single weightings depends on the clustering method. The goal is to find out which prosodic features can be ignored and which are very important. Interesting is, that the significance of the feature weighting also depends on the clustering algorithm. For example in the experiments (#3, #9, #38, #39) *weighting 1* achieves better and worse development results than *weighting 0*.

| # | Method | Distance | #Wgt | #Clust | Dev score | ACRR | TMCL | Test score |
|---|---|---|---|---|---|---|---|---|
| 56 | Bottom up | Furtheat | 5 | 20 | 86.2 | 64.5 | 19.5 | 45.0 |
| 13 | Bottom up | Furthest | 2 | 20 | 64.7 | 63.6 | 23.7 | 39.9 |
| 3 | Bottom up | Furthest | 0 | 20 | 77.4 | 60.9 | 24.9 | 36.0 |
| 10 | Bottom up | Furthest | 2 | 20 | 18.0 | 66.0 | 31.2 | 34.8 |
| 64 | Split &Merge | Furthest | 6 | 20 | 88.7 | 66.9 | 32.4 | 34.5 |
| 61 | S&M + k-Means | Furthest | 5 | 20 | 82.7 | 56.3 | 22.4 | 33.9 |
| 55 | Bott.up + k-Means | Furthest | 3 | 20 | 85.0 | 57.9 | 27.8 | 30.1 |
| 41 | S&M + k-Means | Furthest | 3 | 20 | 68.0 | 55.1 | 25.3 | 29.8 |
| 67 | S&M + k-Means | Furthest | 6 | 20 | 72.5 | 49.2 | 19.9 | 29.3 |
| 52 | S&M + k-Means | Furthest | 4 | 20 | 77.1 | 68.6 | 39.2 | 29.2 |

Table 6.2: Top 10 test score experiments using 20 clusters.

Regarding the list of the ten best test score results of experiments using 20 clusters (Table 6.2) *weighting 5*, *weighting 6*, *weighting3*, and *weighting 2* occur twice, while *weighting 0* and *weighting 4* occur once. *Weighting 1* and *weighting 7* do not occur in this list.
In the list of the best five test scores *weighting 5* occurs twice. *Weighting 0*, *weighting 2* and *weighting 6* occur once.
This leads to the conclusion that *weighting 5* is probably the most adequate weighting for this approach.
*Weighting 2*, which ignores the energy features, leads to improvement of the performance compared to no modification of the token distance.
The use of only 0.0 and 1.0 as weighting based on the ratio of the features in 5.2 is established in *weighting 6*. This weighting achieves similar results like *weighting 5*, but latter one still perform a little better. *Weighting 3*, which rates the spectral tilt as high as duration still occurs twice in the top ten result list, although the spectral tilt can be considered as unimportant. This does not have to be a contradiction. In this weighting the most features, which are considered as very important like in *weighting 6* are also weighted very high. *Weighting 5* shows also, that these "unimportant" features still carry some information.
The performance using *weighting 4* is neither bad nor good. It concentrates on the main features figured out in the early approach [14].
Surprising is the insufficient performance of *weighting 7*. This weighting achieves a development score of 102.6 in the experiments with a reduced training set. The results are not any approximating when using the complete training set.
*Weighting 1* is the only weighting which can be clearly considered as inadequate.

More feature weightings should be tested in further approaches. The performance depends on the weighting *and* the cluster method. From the tested weightings *weighting 5* and *weighting 6* achieve the best results. No modification and the ignoring of the energy features is still a suitable way to create a reasonable classifier.

### 6.1.4 Number of Clusters

During this approach the data is clustered into 20 or 40 clusters. 20 and 40 are rather arbitrarily chosen numbers. 20 seems to be a reasonable number, because the visualization of the tokens indicated, that one cluster per class

will be insufficient, especially for the scattered classes "listen" and "interest".
If the number of clusters is increased further, e.g. to 40, the clusters become too specialized. A very high number would probably cause a very high *development score*, but a poor *test score*. This phenomenom is called *over fitting*.

### 6.1.5 Test Score

The *test score* is derived from the classifier performance of the independent *test set*. In contrast to the development set, the classifier has no "knowledge" about the test set. This is why it is the only score to draw fair and honest conclusions from the approach.
But testing always on the same set could cause random peaks in the performance. This is why both scores, the *development score* as close evaluation and the *test score* are considered as very important.
The development shall indicate a good separation of the classes in the development set and the test score evaluates how this separation is transferred to an independent data set. So both scores have to be high.

Figure 6.1 displays the ratio of the test score and the development score. This plot only illustrates results from experiments clustering to 20 clusters. Each point represents one experiment, the coordinates are the development score and the test score. This figure does not show information about the experimental setup.
It seems that the development score corresponds to the test score. A high development score also achieves at least a positive test score, while methods with a low development score also perform insufficient for the test score.
The most outliers from this observation have a low development score. This means that the clusters are absolutely insufficient for the development set, but achieve a higher test score by random. Also a factor for this phenomenom is, that the test score is mostly greater zero, because only the reduced merged classes are tested while the development score also includes the results before merging, which are worse than after the merging.

### 6.1.6 Conclusion

Apparently there are several ways to achieve good clusters. No method singled out most adequate for the clustering. Only *Top Down* can be regarded

as inappropriate for this problem.
The decision of the cluster distance measure is much clearer. Here, the *furthest neighbor* established as the best measurement.
The number of clusters are not tested very much. 20 clusters can be considered as quite reasonable to create a good classifier.
The behavior of the results concerning the weighting seems to be unpredictable. All weightings seem to be adequate in different clustering methods. A clear conclusion is difficult, but weighting 5 and weighting 6 both achieve a very good performance.

The best results performed experiment #56. This is a *Bottom Up* clustering and clusters into 20 clusters. As distance measure, the furthest neighbor and weighting 5 is used. The average class recognition rate is 64.5% and the token misclassification is 19.5%. In total this is a test score of 45.0%.
The confusion matrix of this experiment can be found on page 53.

## 6.2 "honma" Experiments & Results

A part of this approach is also to analyze the portability of the results from the "un" experiments to other backchannels. The same experimental setups are used on the "honma" data which are introduced in section 5.1.4. The problem in this case is, that neither a similar data set, nor a similarity between the labels exists. 14 different classes in a set of 393 tokens is too few to draw any reasonable conclusions.

The same software and the same experimental setups are used for the experiments, but on two little differing training sets. Both sets contain the developments set, but one also consists of the unlabeled tokens, which have a poor recording quality.
The development score showed, that clustering only the development set achieves a higher performance than the additional use of the unlabeled data. In this section only the experiments without the unlabeled data will be discussed.

The test score on the honma experiments is incredibly low. Only 13 out of 72 experiments achieves a correct vector classification over 10.0%. Experiment *#15* and experiment *#43* achieve the best performance. First one is a

| | imp | dsg | mix | lgh | dsp | ok | qst | uds | bc | stf | uhp | srp | hpp | dbt |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| impress | 1 | 2 | 3 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 12.5 | 25.0 | 37.5 | 12.5 | 12.5 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| disgust | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 0.0 | 50.0 | 0.0 | 0.0 | 50.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| mixed | 1 | 6 | 8 | 1 | 2 | 0 | 1 | 1 | 0 | 0 | 6 | 3 | 0 | 0 |
| | 3.4 | 20.7 | 27.6 | 3.4 | 6.9 | 0.0 | 3.4 | 3.4 | 0.0 | 0.0 | 20.7 | 10.3 | 0.0 | 0.0 |
| laugh | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| | 0.0 | 0.0 | 0.0 | 50.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 50.0 | 0.0 | 0.0 | 0.0 |
| disappo | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 50.0 | 50.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| ok | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 0.0 | 20.0 | 20.0 | 0.0 | 0.0 | 20.0 | 20.0 | 20.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| questio | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 50.0 | 0.0 | 0.0 | 0.0 | 16.7 | 16.7 | 16.7 | 0.0 |
| underst | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 2 | 0 | 1 |
| | 0.0 | 12.5 | 12.5 | 12.5 | 0.0 | 0.0 | 12.5 | 0.0 | 0.0 | 12.5 | 0.0 | 25.0 | 0.0 | 12.5 |
| bc | 0 | 3 | 2 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 5 | 2 | 0 | 0 |
| | 0.0 | 20.0 | 13.3 | 0.0 | 0.0 | 0.0 | 6.7 | 0.0 | 6.7 | 6.7 | 33.3 | 13.3 | 0.0 | 0.0 |
| satisfi | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 100.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| unhappy | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| | 33.3 | 0.0 | 33.3 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 33.3 | 0.0 | 0.0 | 0.0 |
| surpris | 0 | 0 | 2 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| | 0.0 | 0.0 | 33.3 | 0.0 | 0.0 | 0.0 | 33.3 | 0.0 | 0.0 | 0.0 | 0.0 | 16.7 | 16.7 | 0.0 |
| happy | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 3 | 0 | 0 |
| | 18.2 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 54.5 | 27.3 | 0.0 | 0.0 |
| doubtfu | 1 | 0 | 0 | 1 | 0 | 1 | 4 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| | 12.5 | 0.0 | 0.0 | 12.5 | 0.0 | 12.5 | 50.0 | 0.0 | 0.0 | 0.0 | 0.0 | 12.5 | 0.0 | 0.0 |

Table 6.3: Confusion matrix of the "honma" classifier. Upper row is the absolute frequency and lower row the percentage of vectors in each class.

*Bottom Up* clustering, using the *furthest neighbor* as distance measure and *weighting 1* as feature weighting. It achieves an averaged class recognition rate of 18.8% and classifies 18 (17.0%) vectors correctly.
Similar results in experiment *#43*. A combination of *Split & Merge* and *k–Means* algorithm using *weighting 1* perform an averaged class recognition rate of 19.1%. Again 18 (17.0%) vectors are classified correctly.
In six out of these 13 "good" experiments, *weighting 1* is used. No weighting is used five times, while *weighting 4* and *weighting 6* perform once better than 10.0%.
The best performance is achieved in experiment *#43*. The confusion matrix of this clustering is illustrated in Table 6.3. No clear conclusions are derivable from this matrix. The result is better than random guessing, but still not sufficient. But it is too early to draw the conclusion that there is no transferability to the "honma" data. Both data sets are too different from each other, that the a fair comparison is not possible.

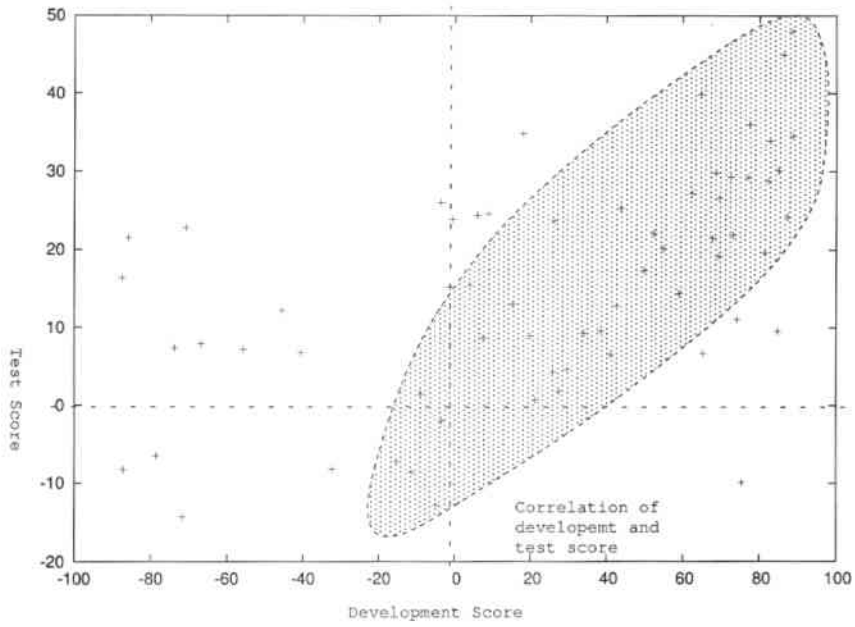


Figure 6.1: Ratio of test score and development score.

# Chapter 7

# SUMMARY

Spontaneous speech is expressive speech and for computers more difficult to understand. The interpretation of *what* is said is not always possible, as long as there is no information about *how* it is said. In Japanese spontaneous speech, the token "un" (comparable to the English "mhm") can have different meanings, like "Yes" and "No".

To make a semantic interpretation of these so called *backchannels* possible, 13 prosodic features based on F0, duration, power, and glottal characteristics have been extracted. The feature vectors have been automatically clustered to build a classifier, which assigns a token to one of several speech act classes.

Features based on F0 are the maximum (F0max), the mean (F0mean), and the minimum (F0min) of the pitch, although the gradient between the minimum and the maximum (F0grad), the location of the maximum (F0pos), and the percentage share of voiced pitch (F0vcd). The power based features are the maximum (Pwrmin), the mean (Pwrmean), the mean (Pwrmean) energy and the location of the maximum (Pwrpos). Also the duration (Dur), the spectral tilt (H1-A3) and the opening quotient of the glottis (H1-H2) have been features in the prosodic feature vectors.

The data was collected for the JST/CREST Expressive Speech Project and is a subset of 150 hours of Japanese everyday speech. The speaker is a Japanese female wearing a microphone and a Minidisk recorder for a whole day. The data consisted of 482 labeled and 2167 unlabeled "un" tokens.
The tokens have been labeled to the following speech acts: "listen", "understand", "interest", "affirm", "affirm overall", "callback", "disagree", "emotion".

Experimental and visual observations led to the conclusion, that the classes "listen" and "understand", "affirm" and "affirm overall", and "disagree" and "emotion" can be merged.

The unlabeled data was used for unsupervised clustering while a subset of the labeled data assigned a speech act class to the clusters to built a classifier. The remaining unused labeled tokens were the test set for a final independent test of the classifier.

The test set consisted of 240 tokens. The classifier classified 198 tokens (82,5%) correctly and achieved an class recognition rate of 65.1%. Table 7.1 shows the confusion matrix of the classifier. The clusters for this classifier were created by a *Bottom Up* clustering, using *furthest neighbor* as distance measure between the clusters and weighted *F0mean, F0pos, F0grad, Pwr-max,* and *Dur* strongly.

| | classified as | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | listen | | interest | | agree | | disagree | |
| class | n | % | n | % | n | % | n | % |
| listen | 181 | 85.8 | 5 | 2.4 | 9 | 4.3 | 16 | 7.5 |
| interest | 3 | 33.3 | 4 | 44.4 | 2 | 22.2 | 0 | 0.0 |
| agree | 2 | 20.0 | 1 | 10.0 | 7 | 70.0 | 0 | 0.0 |
| disagree | 4 | 40.0 | 0 | 0.0 | 0 | 0.0 | 6 | 60.0 |

Table 7.1: Confusion matrix on test set. Absolute number (n) and percentage of vectors in each class is given.

The possibility to extend this approach to other backchannels was analyzed. A dataset of "honma" tokens have also been provided. But as this set consists of only 315 labeled tokens assigned to one of 14 speech act classes, it is too small to be an adequate database for clustering and classification. The lack of data and the high number of classes inhibited to create a sufficient classifier.

This research has shown that the interpretation of backchannels is possible. In the future automatic speech recognizers will be able to understand the meaning of what is said and will interpret emotions of the speaker and the listener. This progress allows the interpretation of the state of mind of the speaker and a semantic analysis of a dialog. Now an universal speech recognizer is one step closer.

# FUTURE WORK

For future approaches more labeled data is needed. A statistical approach using HMMs or GMMs could become possible and further improve the classification results. An approach using neural networks could also be considered. The database consists of only one single speaker. Further research should include more speakers to create a speaker independent classification.
A research considering the covariances between features and dynamic feature vectors could improve the accuracy of the token distance measure and so further improve the classifier.

# ACKNOWLEDGMENT

The research reported here was supported in part by a contract with the Telecommunications Advancement Organization of Japan entitled, "A study of speech dialogue translation technology based on a large corpus".

We want to thank JST/CREST ESP for kindly providing the data and especially Nick Campbell for valuable discussions.

We also thank Prof. Alex Waibel from the University of Karlsruhe in Germany to make this internship and this research possible.

Thanks also goes to Hartwig Holzapfel, my supervisor in Karlsruhe.

I say "Thank You" to all members of the ATR SLT department 1 for giving advice and fair comments, especially Satoshi Nakamura, Konstantin Markov, Frank Soong, and Tobias Cincarek.

Especially I want to thank Rainer Gruhn, who has been a patient supervisor of this research and a good friend and companion during my stay in Japan.

# Bibliography


[1] A. Batliner, A. Kießling, S. Burger, E. Nöth. Filled pauses in spontaneous speech. In *Proc. XIIIth ICPhS*, pages 472–475, 1995.

[2] A. Kießling, R. Kompe, H. Niemann, E. Nöth. DP-Based Determination of F0 Contours from Speech Signals. In *ICASSP*, 1992.

[3] A. Kießling, R. Kompe, H. Niemann, E. Nöth. Detection of Phrase Boundaries and Accents. In *Progress and Prospects of Speech Research and Technology*, 1994.

[4] N. Campbell. Databases of Expressive Speech. In *Proc ISCA (International Speech Communication and Association) ITRW in Speech and Emotion*, pages 34–38, 2000.

[5] N. Campbell. Recording techniques capturing natural every-day speech. In *Proc LREC*, 2002.

[6] JST/CREST Expressive Speech Processing project. introductory web pages at: www.isd.atr.co.jp/esp.

[7] Kerstin Fischer. Die Ikonizität der Pause: Zwischen kognitiver Last und kommunikativer Funktion. In *Proceedings der 4. Fachtagung der Gesellschaft für Kognitionswissenschaft*, 1999.

[8] H. Fujisaki. Prosody, models, and spontaneous speech. In N. Higuchi Y. Sagisaka, N. Campbell, editor, *Computing Prosody*, chapter 3, pages 27–42. Springer, 1996.

[9] Helen M. Hanson. Glottal characteristics of female speakers: Acoustic correlates. In *Acoustical Society of America*, 1997.

[10] LDC Linguistic Data Consortium. CALLHOME. introductory web pages at: www.ldc.upenn.edu.

[11] I. Rogina T. Schulz. Acoustic and language modeling of human and nonhuman noises for human-to-human spontaneous speech recognition. In *Proc. ICASSP* , 1995.

[12] Thomas S. Polzin, Alex H. Waibel. Detecting emotions in speech. In *Proceedings of the CMC*, 1998.

[13] Junji Umeno. Analysis of Para-linguistic and Non-linguistic Information Based on Acoustic Parameters of Aizuchi – 'un' in the Conversation. Master's thesis, Nara Institute of Science and Technology, Nara, 2003.

[14] W. Svojanovsky, R. Gruhn, S. Nakamura. Classification of Nonverbal Utterances In Japanese Spontaneous Speech. In *Information Processing Society of Japan*, pages 277–278, 2004.

[15] A. Waibel. *Prosody and Speech Recognition.* Pitman, 1988.

# Appendix A

# Experimental Setups and Results

| # | Method | Distance | #Wgt | #Clust | Dev score | ACRR | TMCL | Test score |
|---|---|---|---|---|---|---|---|---|
| 1 | Bottom Up | Center | 0 | 20 | 4.0 | 49.9 | 34.4 | 15.5 |
| 2 | Canceled | - | - | - | - | - | - | - |
| 3 | Bottom Up | Furthest | 0 | 20 | 77.4 | 60.9 | 24.9 | 36.0 |
| 4 | Bottom Up | Average | 0 | 20 | -87.3 | 43.2 | 51.5 | -8.3 |
| 5 | Bottom Up | Represent | 0 | 20 | -87.6 | 60.0 | 43.6 | 16.4 |
| 6 | Bottom Up | Center | 1 | 20 | -1.4 | 57.9 | 42.7 | 15.2 |
| 7 | Bottom Up | Average | 1 | 20 | -3.6 | 55.9 | 29.9 | 26.0 |
| 8 | Bottom Up | Represent | 1 | 20 | -70.9 | 58.9 | 36.1 | 22.8 |
| 9 | Bottom Up | Furthest | 1 | 20 | 84.7 | 53.2 | 43.6 | 9.6 |
| 10 | Bottom Up | Center | 2 | 20 | 18.0 | 66.0 | 31.2 | 34.8 |
| 11 | Bottom Up | Average | 2 | 20 | -78.7 | 56.6 | 63.1 | -6.5 |
| 12 | Bottom Up | Represent | 2 | 20 | -3.6 | 61.5 | 63.5 | -2.0 |
| 13 | Bottom Up | Furthest | 2 | 20 | 64.7 | 63.6 | 23.7 | 39.9 |
| 14 | Bottom Up | Furthest | 0 | 40 | 90.0 | 59.6 | 29.4 | 30.2 |
| 15 | Bottom Up | Furthest | 1 | 40 | 95.1 | 53.3 | 52.3 | 1.0 |
| 16 | Bottom Up | Furthest | 2 | 40 | 43.4 | 66.7 | 30.3 | 36.4 |
| 17 | Top Down | Fast | 0 | 20 | -71.8 | 42.1 | 56.4 | -14.3 |
| 18 | Top Down | Furthest | 0 | 20 | -32.4 | 51.1 | 59.3 | -8.2 |
| 19 | Top Down | Fast | 1 | 20 | -0.6 | 61.2 | 37.3 | 23.9 |
| 20 | Top Down | Furthest | 1 | 20 | 6.0 | 59.3 | 34.9 | 24.4 |
| 21 | Top Down | Fast | 2 | 20 | -66.8 | 55.6 | 47.7 | 7.9 |
| 22 | Top Down | Furthest | 2 | 20 | -73.8 | 55.5 | 48.1 | 7.4 |
| 23 | Split & Merge | Furthest | 0 | 20 | -55.8 | 60.3 | 53.1 | 7.2 |
| 24 | Split & Merge | Furthest | 1 | 20 | -45.6 | 57.0 | 44.8 | 12.2 |
| 25 | Split & Merge | Furthest | 2 | 20 | 62.5 | 64.5 | 37.3 | 27.2 |
| 26 | k-Means | Random 1 | 0 | 20 | 25.9 | 54.1 | 49.8 | 4.3 |
| 27 | k-Means | Random 2 | 0 | 20 | 8.8 | 54.4 | 29.8 | 24.6 |
| 28 | k-Means | Random 3 | 0 | 20 | 38.5 | 47.8 | 38.2 | 9.6 |
| 29 | k-Means | Random 1 | 1 | 20 | 27.6 | 50.8 | 49.0 | 1.8 |
| 30 | k-Means | Random 2 | 1 | 20 | -4.9 | 46.1 | 58.9 | -12.8 |
| 31 | k-Means | Random 3 | 1 | 20 | 54.9 | 62.9 | 42.7 | 20.2 |
| 32 | k-Means | Random 1 | 2 | 20 | -11.4 | 58.6 | 67.2 | -8.6 |
| 33 | k-Means | Random 2 | 2 | 20 | 21.2 | 55.1 | 54.4 | 0.7 |
| 34 | k-Means | Random 3 | 2 | 20 | 7.5 | 58.5 | 49.8 | 8.7 |
| 35 | Bott.Up + kMeans | Furthest | 0 | 20 | 87.4 | 56.2 | 32.0 | 24.2 |
| 36 | Bott.Up + kMeans | Furthest | 1 | 20 | 34.0 | 52.9 | 43.6 | 9.3 |
| 37 | Bott.Up + kMeans | Furthest | 2 | 20 | 67.8 | 53.5 | 32.0 | 21.5 |
| 38 | S&M + k-Means | Furthest | 0 | 20 | 74.1 | 61.3 | 50.2 | 11.1 |
| 39 | S&M + k-Means | Furthest | 1 | 20 | 49.9 | 66.0 | 48.6 | 17.4 |
| 40 | S&M + k-Means | Furthest | 2 | 20 | 82.2 | 59.5 | 30.7 | 28.8 |
| 41 | S&M + k-Means | Furthest | 3 | 20 | 68.6 | 55.1 | 25.3 | 29.8 |
| 42 | S&M + k-Means | Furthest | 0 | 40 | 58.1 | 52.8 | 44.8 | 8.0 |
| 43 | S&M + k-Means | Furthest | 1 | 40 | 85.0 | 60.8 | 31.5 | 29.3 |
| 44 | S&M + k-Means | Furthest | 2 | 40 | 20.6 | 50.2 | 51.9 | -1.7 |
| 45 | S&M + k-Means | Furthest | 3 | 40 | 26.7 | 62.1 | 54.8 | 7.3 |
| 46 | Bottom Up | Furthest | 3 | 20 | 73.1 | 55.9 | 34.0 | 21.9 |
| 47 | Bottom Up | Furthest | 3 | 40 | 55.5 | 32.4 | 36.9 | -4.5 |
| 48 | Split & Merge | Furthest | 3 | 20 | 29.6 | 50.2 | 45.6 | 4.6 |
| 49 | Bottom Up | Furthest | 4 | 20 | 69.5 | 56.5 | 29.9 | 26.6 |
| 50 | Bottom Up | Furthest | 4 | 40 | 35.0 | 50.6 | 34.9 | 15.7 |

| # | Method | Distance | #Wgt | #Clust | Dev score | ACRR | TMCL | Test score |
|---|---|---|---|---|---|---|---|---|
| 51 | Split & Merge | Furthest | 4 | 20 | 42.5 | 49.3 | 36.5 | 12.8 |
| 52 | S&M + k-Means | Furthest | 4 | 20 | 77.1 | 68.6 | 39.4 | 29.2 |
| 53 | S&M + k-Means | Furthest | 4 | 40 | 15.6 | 56.8 | 54.7 | 2.1 |
| 54 | Bott.Up + kMeans | Furthest | 4 | 20 | 52.4 | 52.0 | 29.9 | 22.1 |
| 55 | Bott.Up + kMeans | Furthest | 3 | 20 | 85.0 | 57.9 | 27.8 | 30.1 |
| 56 | Bottom Up | Furthest | 5 | 20 | 86.3 | 64.5 | 19.5 | 45.0 |
| 57 | Top Down | Furthest | 5 | 20 | -40.6 | 55.8 | 49.0 | 6.8 |
| 58 | Split & Merge | Furthest | 5 | 20 | -9.0 | 58.4 | 56.9 | 1.5 |
| 59 | k-Means | Random 3 | 5 | 20 | 59.0 | 50.5 | 36.1 | 14.4 |
| 60 | Bott.Up + kMeans | Furthest | 5 | 20 | 43.8 | 62.2 | 36.9 | 25.3 |
| 61 | S&M + k-Means | Furthest | 5 | 20 | 82.7 | 56.3 | 22.4 | 33.9 |
| 62 | Bottom Up | Furthest | 6 | 20 | -15.4 | 47.2 | 54.4 | -7.2 |
| 63 | Top Down | Furthest | 6 | 20 | -85.9 | 56.8 | 35.3 | 21.5 |
| 64 | Split & Merge | Furthest | 6 | 20 | 88.7 | 66.9 | 32.4 | 34.5 |
| 65 | k-Means | Random 3 | 6 | 20 | 19.9 | 54.2 | 45.2 | 9.0 |
| 66 | Bott.Up + kMeans | Furthest | 6 | 20 | 81.4 | 50.8 | 31.2 | 19.6 |
| 67 | S&M + k-Means | Furthest | 6 | 20 | 72.5 | 49.2 | 19.9 | 29.3 |
| 68 | Bottom Up | Furthest | 7 | 20 | 69.3 | 53.2 | 34.0 | 19.2 |
| 69 | Top Down | Furthest | 7 | 20 | 75.4 | 54.0 | 63.9 | -9.9 |
| 70 | Split & Merge | Furthest | 7 | 20 | 65.2 | 48.6 | 41.9 | 6.7 |
| 71 | k-Means | Random 3 | 7 | 20 | 15.2 | 57.0 | 44.0 | 13.0 |
| 72 | Bott.Up + kMeans | Furthest | 7 | 20 | 40.9 | 55.6 | 49.0 | 6.6 |
| 73 | S&M + k-Means | Furthest | 7 | 20 | 26.2 | 51.9 | 28.2 | 23.7 |

Table A.1: Overview of all experimental setups and results. Experiment 2 was canceled.

# Appendix B

# Software Annotations

The software for this research was completely written in *Python 2.3*. For audio processing the free *Snack Sound Toolkit 2.2.3* was used.

The appending CD does not contain the upper software, but directories, which contains this thesis, a technical report, the experiments and the experiment software.
The following description lists the directories on the CD and explains their content.

**experiments:** All files needed and created during the experiments can be found in this directory. It contains the directories *un* and *honma*, which lead to the experiment files.

**software:** 5 files are in this directory: *SubCluster.py, pros.py, DataVector.py, Cluster.py* and *pros.setup*.
This is the complete basic software needed for the experiments. A detailed documentation can be found in the technical report.

**techreport:** This directory contains the file *docu.ps* which is a short report of the research and the results. It also contains a detailed documentation of the software used during this approach.

**thesis:** This thesis and the appendant TEX–Files are located in this directory.