Automated Annotation of Political Speech Recordings

Mathias Rask[†]

Working Paper
This version: January 29, 2025

Recent advances in political science have revealed that audio recordings offer a wealth of politically relevant information beyond what can be gleaned from text alone. However, despite the widespread availability of political speech recordings, incorporating audio data into applied research is challenged by incomplete, inaccurate, or even missing annotations. Annotations, such as timestamps marking the start and end of segments and speaker identities, are essential for preprocessing speech audio into distinct units like speeches, utterances, or words, typical units used when studying speech text. In this paper, I develop a deep learning-based annotation pipeline capable of automatically annotating speech recordings with timestamps and speaker identities at the speech level. The pipeline combines speaker diarization, automatic speech recognition, and speaker recognition and requires no prior human-annotated data. This new weakly supervised learning approach for speaker recognition makes it possible to identify speakers without manually compiling reference segments and does not require retraining when new speakers are targeted. I validate the pipeline using recordings from parliamentary debates in the Danish Parliament, demonstrating that the automated annotations are on par with human benchmarks. The pipeline is implemented as open-source Python software, `speechannote`, for broad accessibility.

[†]Corresponding author. PhD Student, Department of Political Science, Aarhus University

1 Introduction

In recent decades, the quantitative analysis of political speeches has become a widely used methodology in political science across subfields. This development is closely tied to the increasing availability and use of digitized text data of political speech (see, for example, Grimmer et al. 2022), which has transformed research agendas in areas such as comparative politics, international relations, and legislative studies. In addition, recent advancements in political methodology have demonstrated that audio data – recordings of political speeches from settings like parliamentary debates, campaign events, and diplomatic meetings – convey a wealth of politically relevant information, including emotional arousal and accent, that goes beyond what can be extracted and gleaned from text alone (Dietrich et al. 2019; Neumann 2019; Knox and Lucas 2021; Cochrane et al. 2022; Tarr et al. 2022; Rittmann 2023; Damann et al. 2024).

Despite the promises of audio data, its use in applied research is hindered by the frequent or total lack of accurate and complete annotations in archives of audio recordings. Audio annotations such as timestamps marking the start and end of segments and speaker identities are essential in preprocessing speech audio into distinct units like speeches, utterances, or words. These units are common in studies of speech text and constitute observations in the audio data. For instance, analyzing the government-opposition divide or the degree of legislative conflict on a policy issue using speech audio requires precise knowledge of when each speech begins and ends, the identity of the corresponding speakers (to determine their partisanship or government status), and possibly the text of each speech (to determine the policy issue). While manual annotation may suffice for small corpora of speech audio, it quickly becomes impractical given the thousands of hours of audio recordings available to research across speech settings such as parliamentary debates in national legislatures or recordings of meetings in different EU institutions.

In this paper, I develop an annotation pipeline based on deep learning (DL) models that can automatically annotate audio recordings of political speech with timestamps, speaker identity, and text at the level of each speech. The pipeline combines tasks from computer science such as speaker

diarization (SD), automatic speech recognition (ASR), and speaker identification (SI) – a variant of speaker recognition (SR) – to construct an end-to-end workflow that simultaneously diarizes a recording into its distinct speeches, identifies speakers' identities, and transcribes the words spoken in speeches. All tasks rely on open-source, pretrained, and state-of-the-art neural network architectures from either the `pyannoate.audio` toolkit (Bredin 2023; Plaquet and Bredin 2023) or OpenAI's SR model, `Whisper` (Radford et al. 2023). Uniquely, the pipeline does not require prior human-annotated data or retraining when new speakers are targeted due to a new weakly supervised learning setup to SI developed in the paper. This method exploits the fact that target speakers can often be located in an audio recording with a corresponding pre-existing transcript. Even if the recording to be annotated does not have a corresponding transcript, reference segments can still be compiled automatically with the weakly supervised setup whenever the target speaker is located in a recording-transcript pair. When the recording to be annotated has a corresponding pre-existing transcript, this can be done in one shot.

The pipeline is validated using a set of audio recordings from parliamentary debates in the Danish Parliament, the Folketing. Using a sample of 21 recordings, I showcase that automated annotations of timestamps and speaker identities are fully on par and comparable with human annotations, showing virtually no differences in downstream measurements of acoustic features such as vocal pitch. Furthermore, I show that the weakly supervised learning approach to SR can compile reference segments for target speakers with the same accuracy as humans and infer the identity of speakers with high precision and recall simultaneously. The automated annotation pipeline can be used to compile comprehensive large-scale datasets of political speech audio that make it possible to study what politicians say and how they say it across speech settings and languages. Open-source Python software, `speechannote`, will be available to implement the pipeline.[1]

[1]The `speechannote` package is not ready for public release in its current state but is expected to be made available at GitHub in early to mid-2025.

2 The Role of Annotations

In recent years, political methodology has increasingly turned towards machine and deep learning (ML/DL) tools to study political behavior in large collections of unstructured data (de Slegte et al. 2024), such as text (e.g., Slapin and Proksch 2008; Herzog and Benoit 2015; Peterson and Spirling 2018; Ash et al. 2024; Castanho Silva et al. 2024), image (e.g., Joo et al. 2019; Williams et al. 2020; Torres and Cantú 2022), and audio (e.g., Rittmann et al. 2020; Knox and Lucas 2021).[2] This turn has focused mostly on measuring where computational methods have been applied to discover, learn, and estimate an unobserved concept from the data. The resulting measure estimate can be seen as a proxy, i.e., a simplifying approximation (Messeri and Crockett 2024, p. 54), that can be used as a variable in a regression model (Knox et al. 2022).

Annotation is as important as measurement, but has received considerably less attention (although, see Tarr et al. 2022). The concept of 'annotation' is rarely explicitly defined and is often used interchangeably with 'labeling' or 'coding'. I define an annotation as information that imposes structure on otherwise unstructured data, such as timestamps marking the start and end of segments and speaker identities denoting who is speaking when.[3] In contrast, labeling and coding often refer to classification tasks where the aim is to assign categories to the data, such as detecting the policy issue in video advertisement (Tarr et al. 2022), classifying a speech as skeptic or non-skeptic (Knox and Lucas 2021), or identifying the race of individuals appearing in images (Anastasopoulos et al. 2024).

Annotations are essential for utilizing audio recordings in applied political science research, as they enable accurate downstream measurements. For example, when calculating utterance-level vocal pitch averages (Dietrich et al. 2019) or speech-level averages of vocal articulation (Neumann 2019), utterance- and speech-level annotations are required to segment the recordings into the ap-

[2]Measures can also be hard-coded by directly extracting textual (e.g., Silva and Proksch 2022), auditory (e.g., Dietrich et al. 2019; Rittmann 2023), or visual features (e.g., Torres 2024) from the text, audio, or image data that can be used as proxies of an underlying unobserved concept.

[3]Note that annotating speaker identities can also be viewed as labeling because the task involves classification. Viewing assigning speaker identities as annotating follows from the definition of annotation.

propriate units of analysis before measuring the auditory features. This process is straightforward when annotations are available, which has been the case in previous research using speech audio. For instance, in studies of Supreme Court justices' voting intentions (Dietrich et al. 2019) and skepticism during oral arguments, measures were obtained at the level of each utterance by segmenting recordings with the help of utterance-level timestamps provided by the Oyez Project.[4] Similarly, in analyses of emotional engagement by legislators in the US House of Representatives (Dietrich et al. 2019) and the German Bundestag (Rittmann 2023), measures were obtained at the speech level by segmenting recordings using timestamps from closed captioning information available at `http://houselive.gov/` and by leveraging the hierarchical structure of the `www.bundestag.de/mediathek` to download individual speeches.[5]

However, most archives do not provide such annotations. In the following three archives of parliamentary debates, annotations are often inaccurate, incomplete, or absent. In the Danish parliament, annotations are inaccurate. Recordings of parliamentary debates include speech-level annotations, but closer inspection reveals frequent errors.[6] In the UK House of Commons, annotations are often incomplete. Recordings of parliamentary question hours are fully and accurately annotated at the speech level, but recordings on legislative debates are incomplete. Lastly, annotations are absent in the Irish Dáil Éireann, making the archive extremely difficult to use for applied research.[7]

The problem of inaccurate, incomplete, or missing annotations extends beyond parliamentary debates. Annotations are often absent in recordings of parliamentary committee proceedings (Kappos 2024), campaign debates between party leaders (Proksch et al. 2019), and public local government meetings (Barari and Simko 2023). This issue is more pronounced in multi-speaker record-

[4] `https://www.oyez.org/`.

[5] Note that the `http://houselive.gov/` archive has since been replaced by `https://live.house.gov/`, which does not allow download of recordings or contain closed captioning with timestamps.

[6] A comparison with a human benchmark shows a timestamp error rate of 40.2%. An example of a recording can be found here: `https://www.ft.dk/aktuelt/webtv/video/20151/salen/75`.

[7] An example of a recording from the Irish Dáil Éireann can be found here: `https://www.oireachtas.ie/en/debates/debate/dail/2015-11-19/`.

ings, but it is also present in single-speaker content, such as campaign videos from YouTube or social media platforms. For example, when analyzing US senators' campaign videos on YouTube, Neumann (2019) faced the challenge of filtering out non-speech elements, such as music, from actual speech. This type of recording also requires precise timestamps to mark the start and end of speech segments to distinguish between speech and non-speech sections.

When the corpus of speech audio is relatively small, segmentation can be done manually by hand-coding timestamps and speaker identities for each unit of analysis. However, manual annotation quickly becomes impractical as the volume of digitized political speech recordings grows, often spanning not just minutes but hundreds or even thousands of hours. This scale necessitates a computational solution that can automatically or semi-automatically annotate speech recordings to enable the use of audio in applied research. In the next section, I outline how tasks from computer science can address this challenge.

3 Computational Annotation of Speech Audio

Several tasks within computer science concern annotating unstructured data. The most prominent tasks regarding speech audio include speaker diarization (SD), automatic speech recognition (ASR), and speaker recognition (SR). In the following, I briefly outline SD and SR and use this as building blocks for the annotation pipeline. ASR has been covered extensively in previous work (Proksch et al. 2019; Tarr et al. 2022; Landesvatter et al. 2023).

3.1 Speaker Diarization (SD)

SD refers to segmenting an audio recording into separate speech segments and assigning each segment to a particular speaker label (Park et al. 2022). In popular terms, it is used to answer the question "who spoke when", typically in multi-speaker recordings where we want to identify the occurrence and boundary of each speaker turn in an unsupervised manner. State-of-the-art systems contain three steps: (1) detecting speech segments using pretrained DL models; (2) generating speaker embeddings for each segment; (3) clustering the embeddings to track segments from the same speakers. The output is a set of speech segments each with a pair of timestamps and a

generic speaker label. This is often concatenated to a higher level with consecutive segments from the same speaker forming one larger segment. The raw and concatenated output is illustrated in Table 1. Since any state-of-the-art SD system operates unsupervised, the speaker labels do not recognize the actual identities of speakers but only keep track of different vocal characteristics through generic labels such as 'A', 'B', 'C', and so on. Hence, the main output from SD is the speech-level timestamp annotations.

A) Raw				**B) Concatenated**				**C) Augmented**
Segment	Start	End	Speaker	Segment	Start	End	Speaker	Speaker
1	5.432	7.610	B	1	5.432	16.502	B	John
2	8.978	16.502	B					
3	20.786	30.002	A	2	20.786	58.197	A	Jane
4	31.111	45.731	A					
5	46.654	58.197	A					
6	59.640	70.112	B	3	59.640	70.112	B	John
7	72.059	80.385	C	4	72.059	89.870	C	Jan
8	83.388	87.500	C					
9	87.905	89.870	C					
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Table 1: Raw, concatenated, and augmented SD output. The concatenated output aggregates the segments to the speech level using a greedy concatenation approach where consecutive segments from the same speaker label are grouped. The augmented output is produced at the concatenated segments applying SR on each segment.

3.2 Speaker Recognition (SR)

SR 'verifies' or 'identifies' a speaker's identity based on voice characteristics. Efforts to recognize speakers date back several decades (Pruzansky and Mathews 1964) and have often been used to augment ASR output with speaker names (Furui 2005). SR can be done as a standalone task where speakers are recognized in single-speaker recordings. However, it is also commonly used to augment SD output by recognizing the segments' corresponding speaker identities (Table 1C).

Whether SR aims at 'verifying' or 'identifying' depends on the task. If the goal is speaker verification (SV), speakers are targeted in a one-to-one matching scheme (i.e., a binary classification task). If the goal is speaker identification (SI), speakers are targeted in a one-to-many matching scheme (i.e., a multi-classification task). The nature of the task also depends on whether the set of target speakers is open or closed. When closed, the segments to be classified only contain speakers already in the target set. When open, the segments may also contain speakers not in the target set.[8]

Virtually any SR system works as supervised learning and contains the same four steps. (1) Reference audio is compiled for each target speaker to be recognized. (2) The compiled reference audio is encoded with either hard-coded acoustic features, e.g., MFCCs (Tiwari 2010), or higher-order representations, e.g., d-vector or x-vector embeddings (Variani et al. 2014; Snyder et al. 2018; Tumminia et al. 2021), to serve as vocal fingerprints. (3) A model, typically a probabilistic LDA or a cosine scoring model (Peng et al. 2022), learns to discriminate between voice characteristics based on the vocal fingerprints. (4) The trained (i.e., learned) model is used to assign speakers to single-speaker recordings, such as concatenated SD segments.[9] Hence, the main output from SR is speech-level speaker annotations.

4 An Automated Annotation Pipeline

To facilitate computational annotation of audio recordings of political speech, I construct an annotation pipeline that automatically annotates speech recordings with timestamps and speaker identities at the level of each speech. The method combines SD and a new weakly supervised learning approach to SR and makes it possible to align text and audio through ASR-generated speech-level transcripts. The automated annotations can be used to construct speech-level measurements from the audio data that can be exploited in substantive analyses.

[8] Applications in political science are most likely to revolve around open-set speaker identification tasks.

[9] PLDA and classification-based approaches to SR have more trainable parameters than cosine scoring models, which, in theory, should make it superior. Yet, research has shown that cosine scoring models using embeddings as voice encoders are as if not more accurate as PLDA classifiers (Peng et al. 2022).

4.1 Data Preparation

As input, the pipeline takes a single audio recording denoted by $\mathcal{Z}$. When $\mathcal{Z}$ is only available in video format, the video channel is dropped and converted to an audio file in `.wav` format before being forwarded to the pipeline. The pipeline requires that the audio signal is sampled at a rate of 16,000 samples per second and contains monaural sound.[10] Since audio recordings of political speeches often are long and contain multiple speeches and speakers, the recording is divided into equal length batches $k \in \{1,\ldots,K\}$ to reduce the computational costs of the annotation, particularly the diarization stage.[11] I use a batch duration of $d = 600$ seconds (i.e., 10 minutes) as default, meaning that a one-hour recording is converted into $K = 6$ batches.[12] I denote a batched recording as $\widetilde{\mathcal{Z}}$ and a specific batch as $\widetilde{\mathcal{Z}}_k$.[13]

4.2 Stage 1: Diarization

The first stage of the pipeline, SD, is applied to obtain timestamp annotations for the corresponding speech segments in an audio recording. To implement SD, I rely on an open-source and pretrained diarization system from `pyannote.audio` (Bredin et al. 2020; Bredin and Laurent 2021). The system uses an end-to-end workflow based on DL building blocks reaching state-of-the-art performance on multiple datasets (Bredin 2023). The software allows for flexible fine-tuning of the individual modules, but the system can be used off-the-shelf with a high baseline accuracy across settings and languages. This feature is important as it makes the system sustainable across different types of speech recordings, such as campaign and parliamentary debates, and multilingual meaning that it can operate on recordings of speech from multilingual parliaments in, e.g., Canada

[10] The command-line tool `ffmpeg` can down- or upsample an audio recording (i.e., change the sampling rate) and manipulate the number of channels without further ado.

[11] Batching is also used by Lükena et al. (2024) to reduce the computational costs in benchmarking the `Python` package, `MEXCA`, enabling researchers to extract emotional expressions in faces, vocalizations, and words.

[12] If the remainder r is less than 60 seconds, the last batch has a length of $d + r$. If $r > 60$ seconds, the last batch K has a duration of r.

[13] The analyses in this paper are conducted using a Quadro RTX 5000 GPU with 16GB RAM. A $d = 600$ (i.e., 10 minutes) batch takes about 18-20 seconds to be diarized. Without a GPU, this would take around 10-15 times longer. Furthermore, the computational time is expected to increase exponentially for the diarization step, at least if more unique speakers appear the longer the recording (Neumann 2019).

(Parliament of Canada in English / Parlement du Canada in French), Belgium, or the EU.

The steps of the diarization stage are illustrated in Figure 1. After the recording $\mathcal{Z}$ has been batched $\widetilde{Z}$, each batch $\widetilde{\mathcal{Z}}_k$ is forwarded to the SD system returning a total of $\widetilde{N}$ speech segments $\widetilde{S}_i$ for $i \in \{1,\ldots,\widetilde{N}\}$ where $\widetilde{S}_i \subseteq \widetilde{\mathcal{Z}}$ and where $\widetilde{N}_k \geq 0$ denotes the number of segments for batch k such that:

$$\sum_{k=1}^{K} \widetilde{N}_k = \widetilde{N}$$

Each speech segment $\widetilde{S}_i$ also has a corresponding set of annotations containing the timestamps marking the start and end of each segment, the segment's generic speaker label as identified by the SD system, and a batch index. While diarized speaker labels are generic, they are coherent such that the same hypothesized speaker is assigned to the same label. This, however, does not apply when using batching as this makes labels coherent only within but not between batches.[14]

Batching also causes sharp discontinuities in the audio signal, as it might falsely cut a speech into two or more batches. I remove discontinuities caused by the batching by resegmenting the diarization output considering the similarity between the last and first segments in batch k and $k+1$ for $0 \leq k \leq K$. The segments are concatenated if they are sufficiently similar, effectively removing discontinuities caused by the batching. The result is a total of N resegmented speech segments S_i for $i \in \{1,\ldots,N\}$ where $S_i \subseteq \mathcal{Z}$ and where $N \leq \widetilde{N}$.

[14]The consequence of batching is that speaker labels are generally uninformative as speaker "C" in batch k is not necessarily the same as speaker "C" in batch $k+1$.

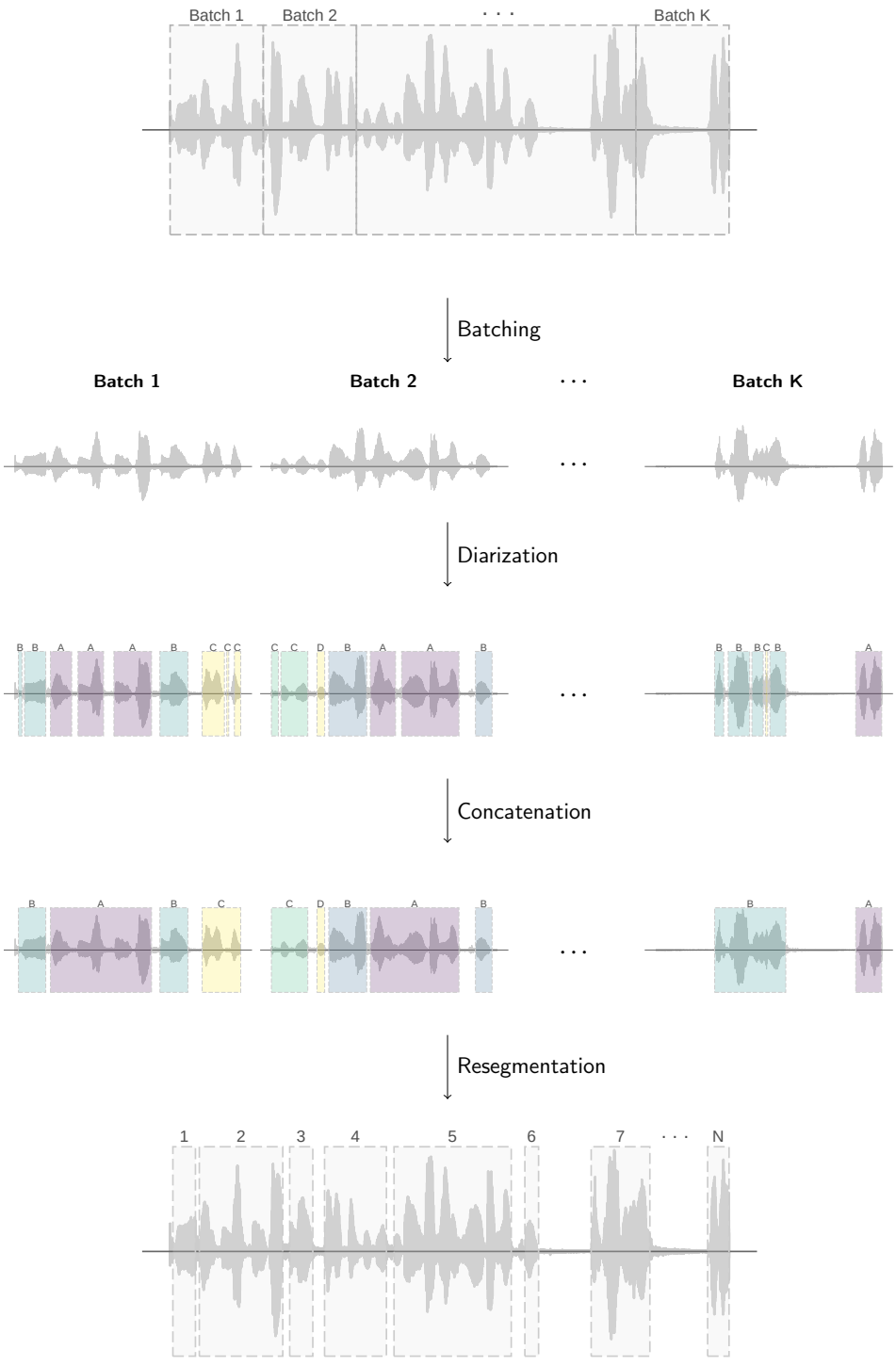


Figure 1: Illustration of diarization stage.

4.2.1 Segment Embeddings

After the diarization output has been resegmented, each speech segment S_i is projected into an $\mathbb{R}^D$ vector space to construct segment embeddings. I denote this by $\mathcal{H} = \{\mathbf{h}_1, \ldots, \mathbf{h}_N\}$. The collection of segment embeddings can be viewed as a $D \times N$ matrix where D is the dimension of the embeddings and N is the number of diarized segments. To compute the embeddings, I rely on pretrained embeddings from pyannote.audio trained with a time delay neural network (TDNN) to generate fixed-length x-vectors from varying-length segments with $D = 512$ (Bredin et al. 2020; Coria et al. 2020).[15]

The segment embeddings are supposed to act as 'voiceprints' (i.e., vocal fingerprints) that encode speakers' voice characteristics. While the embeddings also capture contextual factors such as the type of speech, the speaker's position, or the audience, the dominant variation is the speakers' unique voice characteristics. A useful representation generates embeddings that (1) are close to each other in vector space when belonging to the same speaker, and (2) are distant when belonging to different speakers. That is, the embeddings should exhibit high intra-class and low inter-class similarity.

4.3 Reference Embedding Set

The Reference Embedding Set (`RES`) contains reference segments encoded as D-length reference embeddings for each target speaker $\{1, \ldots, L\}$ to be identified. Each reference segment is extracted using a corresponding set of timestamps and then projected into embeddings denoted as:

$$\mathcal{R} = \{\mathcal{R}^1, \mathcal{R}^2, \ldots, \mathcal{R}^L\} \tag{1}$$

where $\mathcal{R}^l$ denotes the collection of reference embeddings for speaker l. A speaker's collection of reference embeddings $\mathcal{R}^l$ act as reference voiceprints with a total of $U \geq 1$ voiceprints for each speaker:

[15] The fixed-length property means that every segment, no matter the duration, is represented as a 512-dimensional vector.

$$\mathcal{R}^l = \{\mathbf{r}_1^l, \mathbf{r}_2^l, \dots \mathbf{r}_u^l\} \tag{2}$$

where $\mathbf{r}^l \in \mathbb{R}^D$. Similarly to the segment embeddings, the reference embeddings are projected into fixed-length embeddings with $D = 512$ using the pretrained embeddings from pyannote.audio (Bredin et al. 2020; Coria et al. 2020).

4.4 Compiling Reference Embeddings

A conventional SR task compiles annotations of reference segments manually due to the supervised learning setup necessary to recognize speaker identities. This approach works well for small-scale applications where the population of target speakers is small, and the set of target speakers is static, but it does not scale well or offer much flexibility. Often, political scientists are interested in settings with a large set of target speakers and a dynamic set, e.g., when studying MPs elected to a parliament during a T year period (e.g., three parliamentary terms).[16]

To accommodate these issues, I develop a method compiling reference segments using a weakly supervised setup.[17] The method operates through fuzzy string matching where ASR-generated texts computed on speech segments are compared and linked to pre-existing transcript texts to extract information about the speaker's identity. This feature is the crux of the automatism in the pipeline as it facilitates identification of speakers using 'weak references'. The setup rests on the joint availability of audio recordings containing a large number of speeches given by multiple different speakers and corresponding pre-existing speech level transcript (e.g., ParlSpeech V2) containing information about the text and speaker identity at the speech level. Whenever a speaker

[16]For instance, in the June 2024 election to the UK House of Commons, a record 335 new MPs were elected to parliament.https://www.theguardian.com/politics/article/2024/jul/09/record-335-new-mps-to-be-inducted-into-house-of-commons-this-week. In such a case, assuming that each speaker only contains a single reference, 335 new references must be manually compiled before each speaker can be recognized by a classifier.

[17]Weak supervision is a variant of machine learning where labels (i.e., reference audio) are obtained automatically using noisy supervisory signals (e.g., Karu and Alumäe 2018). The inference works exactly as supervised learning setups but differs in how the labels (i.e., the references) are obtained. The 'weak' part of the supervision comes from the fact that labels are not manually verified.

can be located in an audio recording with a corresponding pre-existing transcript, the speaker can be automatically identified in any other speech recording. That is, even if the recording to be annotated does not have a corresponding transcript, reference segments can still be compiled with the weakly supervised setup whenever the target speaker is found in any pre-existing transcript.

The workflow is illustrated in Figure 2. First, an audio recording is segmented, presumably using SD, and subsequently resegmented if batching is applied. The resulting speech segments are then passed to an ASR system that automatically transcribes the words spoken to generate segment texts with $\widehat{W}_i \in \widehat{\mathcal{W}}$ for $i \in \{1,\ldots,N\}$. For this paper, I use OpenAI's open-source, multilingual ASR model Whisper, which has been shown to achieve the lowest Word Error Rate (WER) among a range of models (Landesvatter et al. 2023).[18] I denote the pre-existing transcript texts $\widetilde{W}_j \in \widetilde{\mathcal{W}}$ for $j \in \{1,\ldots,M\}$ and refer to this as the target texts.

The segment $\widehat{\mathcal{W}}$ and target texts $\widetilde{\mathcal{W}}$ can both be viewed as "estimates" of the true spoken words $\mathcal{W}$. Segment texts are an estimate in the sense that while ASR systems are generally accurate, they still commit errors (e.g., Proksch et al. 2019). Target texts are an estimate in the sense that pre-existing transcripts are often non-verbatim, meaning that the spoken words are edited and corrected before being published.[19] This also illustrates why the string matching is fuzzy rather than exact. Segment texts do not match exactly but approximately to target texts due to the errors committed by the ASR model and the non-verbatim transcripts used as target texts.

[18] In their validation study of transcribing voice data from surveys, Landesvatter et al. (2023) find that Whisper achieves the lowest WER among Google Cloud Speech-to-Text, wav2vec 2.0, and Nvidia (NeMo).

[19] For example, the official report of all parliamentary debates from the UK Parliament, Hansard, denotes their transcripts as being "substantially verbatim" referring to the fact speeches are "edited to remove repetitions and obvious mistakes". The Office of the Folketing Hansard, Folketingstidende, also calls their report "substantially verbatim". Before the reports are published, speeches are edited following four general editing guidelines as defined by the Presidium of the Danish Parliament.

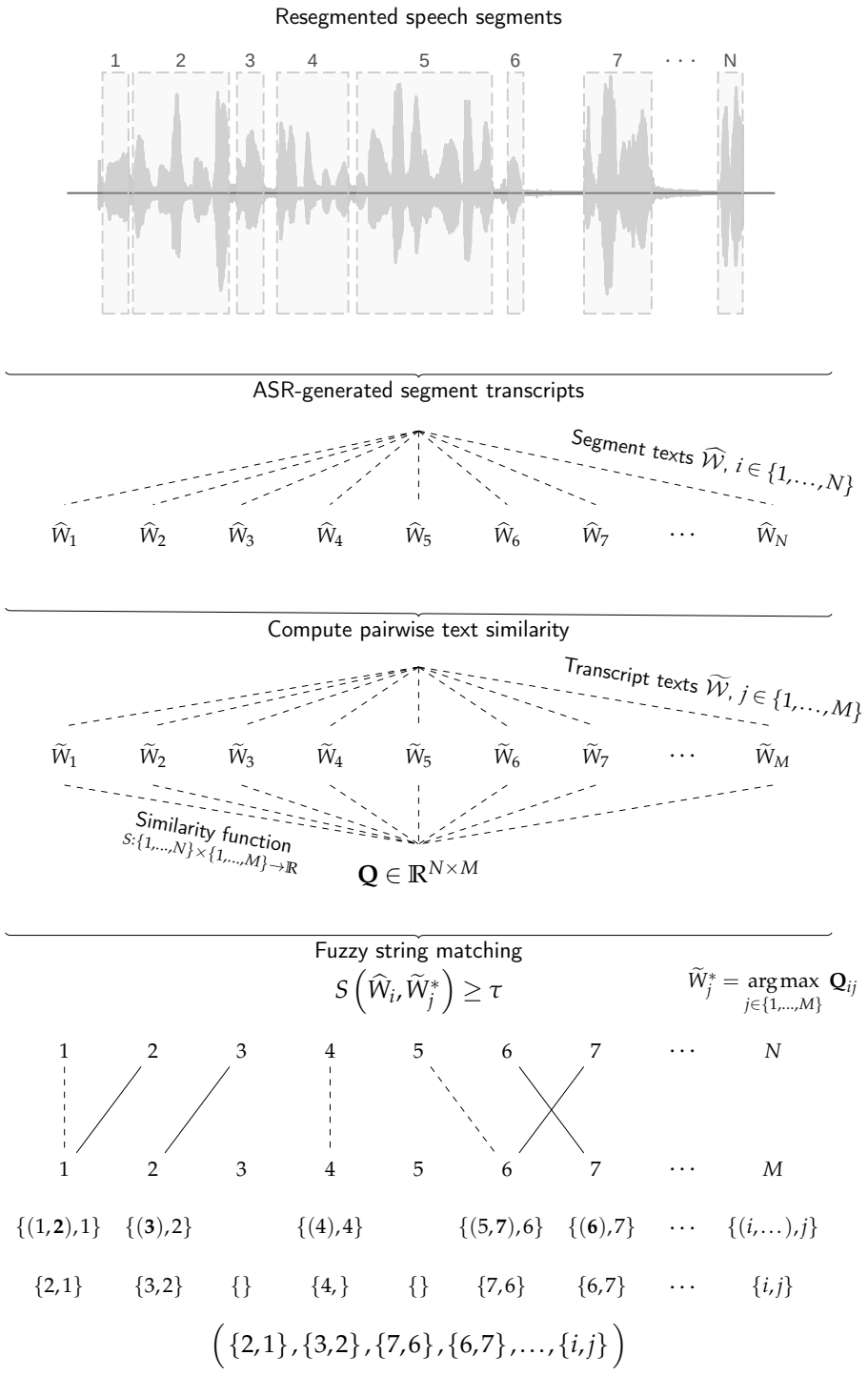


Figure 2: Illustration of compiling reference segments using the weakly supervised learning approach.

Before the texts are matched, $\widehat{\mathcal{W}}$ and $\widetilde{\mathcal{W}}$ are preprocessed and vectorized. As a baseline, texts are only minimally preprocessed, removing punctuation and lower casing. As I will show in the validation analysis, this proves sufficient, and removal of stopwords, lemmatization, or stemming appears to be unnecessary. Before vectorization, the texts are tokenized at the level of each word. Words are used as the unit of tokenization because the texts (i.e., the speeches) appear more like documents than strings, although this increases sensitivity to misspellings generated by the ASR model. As a result, the texts are vectorized as bag-of-words and not bag-of-letters as in other fuzzy string matching tasks (Kaufman and Klevs 2022). While a bag-of-words approach is a simple representation, this also proves sufficient, likely because synonyms and semantically similar words should not be deemed similar in the current task.[20]

After vectorization, the text vectors are transformed into an $N \times M$ similarity matrix $\mathbf{Q} \in \mathbb{R}^{N \times M}$ where the (i,j) entry denotes the similarity between segment text i and target text j. The similarity between texts is computed using the similarity function $S : \{1,\ldots,N\} \times \{1,\ldots,M\} \to \mathbb{R}$ with $S(\mathbf{a},\mathbf{b}) \in [0,1]$ and $\mathbf{a} \geq 0$ and $\mathbf{b} \geq 0$:

$$S(\mathbf{a},\mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \|\mathbf{b}\|} = \frac{\sum_{t=1}^{n} a_t b_t}{\sqrt{\sum_{t=1}^{n} a_t^2} \sqrt{\sum_{t=1}^{n} b_t^2}} \tag{3}$$

which corresponds to the cosine similarity metric where a_t and b_t are the tth component of the n-dimensional vectors $\mathbf{a}$ and $\mathbf{b}$, respectively.[21] The most similar target text to segment text $\widehat{W}_i$ is computed using a 1-nearest-neighbor approach where the nearest neighbor is defined as:

$$\widetilde{W}_j^* = \underset{j \in \{1,\ldots,M\}}{\arg\max}\ \mathbf{Q}_{ij} \tag{4}$$

The nearest neighbor is computed for each segment text, creating a total of N nearest neigh-

[20] Wang (2024) shows that LLMs can be used to also match strings, particularly when semantically similar words and phrases such as "JP Morgan" and "Chase Bank" should be matched.

[21] Note that the metric is lower bounded by zero because the bag-of-words is non-negative.

bors.[22] A segment text is then matched to the nearest target text if it exceeds threshold τ:

$$S\left(\widehat{W}_i, \widetilde{W}_j^*\right) \geq \tau \tag{5}$$

As illustrated in Figure 2, this creates a set of matches that enable extraction of a speaker's identity. Since the pre-existing transcript has speech-level metadata such as the speaker's identity accompanying the non-verbatim speech transcripts, a textual match automatically retrieves a speaker's identity.

4.5 Stage 2: Speaker Inference

The second stage of the pipeline is to assign target speakers to the resegmented speech segments using SR based on the references compiled in `RES`. I implement SR using a cosine scoring model that allows speakers to be flexibly identified and adapted with no re-training required when recognizing new targets (Tumminia et al. 2021). Furthermore, since `RES` likely does not include every speaker recognized by the system, the cosine scoring comes with a threshold hyperparameter λ that enables discrimination between "known" (in `RES`) and "unknown" speakers (not in `RES`). This makes it an open-set task with the set of target speakers being denoted as $\mathcal{Y} = \{1, 2, \ldots, L, L+1\}$ where $L+1$ allows for an "unknown" speaker if the cosine scoring does not exceed the threshold.

To assign speakers to speech segments, I define a similarity function F mapping $\{1, \ldots, N\} \times \{1, \ldots, L, L+1\} \to \mathbb{R}$ with $F(\cdot) \in [-1, 1]$ based on the dot product between two n-dimensional vectors:

$$F(\mathbf{a}, \mathbf{b}) = \mathbf{a} \cdot \mathbf{b} \tag{6}$$

corresponding to the cosine similarity when the n-dimensional vectors have unit norm. The similarity function is lower bounded by -1 and upper bounded by 1, the former indicating that vectors are completely dissimilar, zero indicating they are geometrically orthogonal, and the latter indicat-

[22] If more than one segment text is linked to the same target text, the highest similarity wins the tie.

ing they are completely similar.

The function F is used to compute the similarity between segment $\mathcal{H}$ and reference embeddings $\mathcal{R}$ to determine who the speaker is for each speech segment S_i. A higher value of F indicates that the segment embedding is closer to the reference embedding than lower values, suggesting that a speaker is more likely to be the true speaker in the speech segment. Given a segment embedding $\mathbf{h} \in \mathcal{H}$, I define the nearest reference embedding as:

$$\mathbf{r}^* = \underset{\mathbf{r} \in \mathcal{R}}{\arg\max}\, F(\mathbf{h}, \mathbf{r}) \tag{7}$$

where $\mathbf{r}^*$ is the most similar reference embedding (i.e., the nearest neighbor) for segment embedding $\mathbf{h}$ among all reference embeddings stored in `RES`. For the final inference, I define a mapping function $m : \mathbb{R}^D \rightarrow \{0,1\}^{L+1}$ that takes in a segment embedding $\mathbf{h} \in \mathcal{H}$ and returns a one-hot encoded vector of length $L+1$:

$$m(\mathbf{h}) = \begin{cases} \mathbb{1}(\mathbf{r}^*), & \text{if} \quad F(\mathbf{h}, \mathbf{r}^*) \geq \lambda, \\ [0, \ldots, 0, 1]^{\mathrm{T}} & \text{else.} \end{cases} \tag{8}$$

where $\mathbb{1}(\cdot)$ is an indicator function that returns the assigned speaker if the nearest neighbor is $\geq \lambda$. For example, $\mathbb{1}(\mathbf{r}_2^4) = [0,0,0,1,\ldots,0]^{\mathrm{T}}$ returns the specified one-hot vector when the most similar reference embedding belongs to the second reference embedding for the fourth speaker in `RES`, whereas $[0,\ldots,0,1]^{\mathrm{T}}$ is returned if the nearest neighbor similarity is $< \lambda$. This inference is done for each segment embedding to assign speakers to the speech segment. This is illustrated in Figure 3 where some speech segments are assigned to speakers (e.g., "Jan" and "John") if identified, whereas other segments are assigned an "unknown" label, $L+1$. The λ-threshold is a hyperparameter analog to the *tau* threshold used in the fuzzy string matching that can be optimized to improve speaker inference, balance precision, and recall. In the validation analysis, I investigate in detail how the choice of threshold impacts the inference.

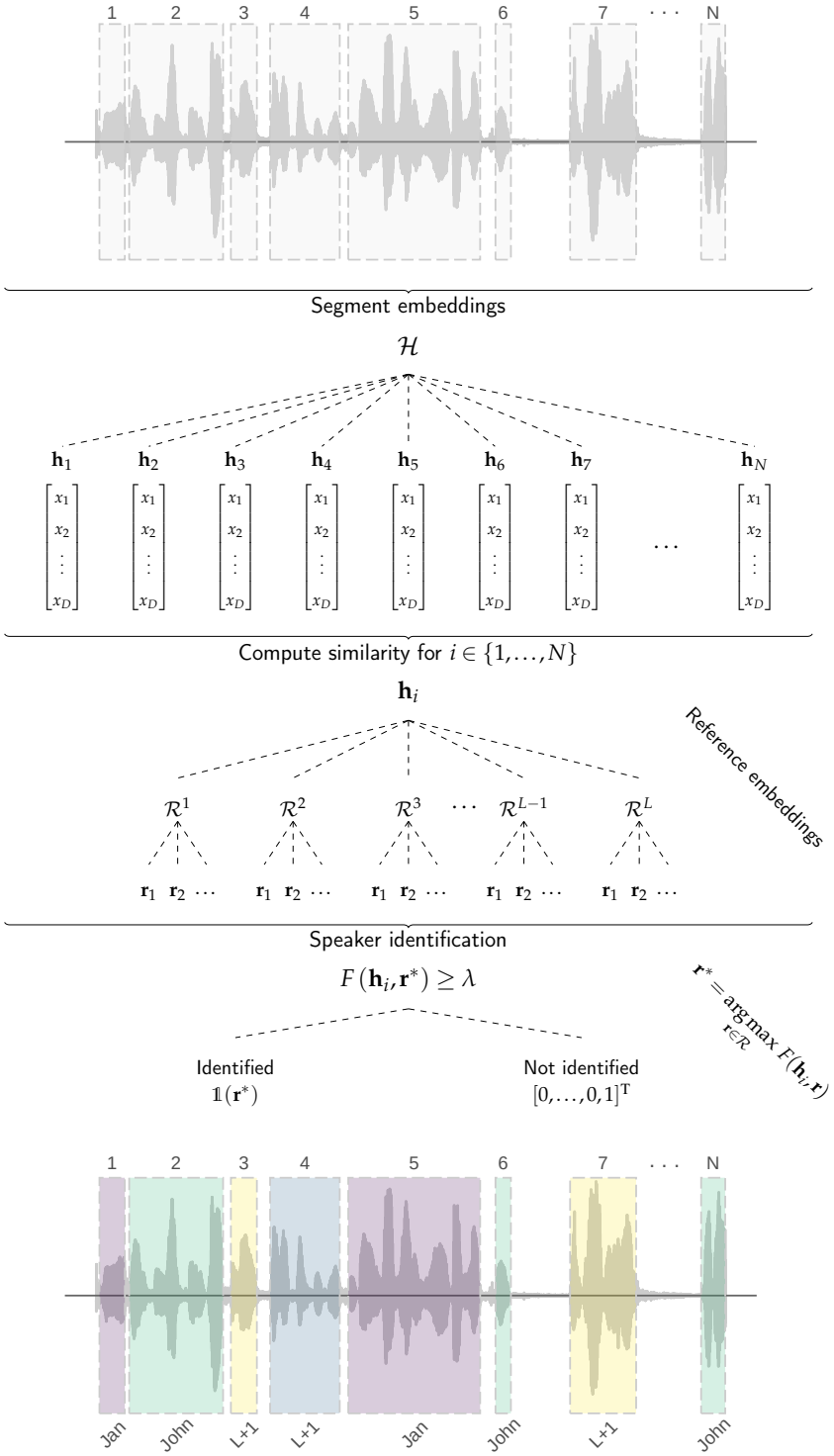


Figure 3: Illustration of inference stage.

5 Validation Analysis

In this section, I evaluate the performance of the individual steps of the automated annotation pipeline using manually annotated audio recordings of parliamentary debates from the Danish Parliament.

5.1 Validation Data

The automated annotations are validated against a human benchmark to score how computationally annotating speech audio compares to manually annotating recordings. For the validation, I focus on recordings of parliamentary debates in the Danish Parliament. The Folketing speaks exclusively Danish, making it a least likely case for validating the pipeline as Danish is not as widespread in ML as, e.g., English, Spanish, or German. Parliamentary debates are an ideal setting to test the validity of the pipeline as they are widely studied in political science (e.g., Back, Debus, and Fernandes 2021), shedding light on topics such as coalition politics, party competition, and political representation. Parliamentary debates also provide a useful boundary case for testing the usability of the pipeline. Compared to other speech settings, parliamentary debates are characterized by a large number of unique speakers taking unequal turns, both in frequency and duration, and their long durability, often more than five hours.

To construct the validation data, I sample a total of 21 parliamentary debates based on the population of debates, defined as those that are jointly covered by the ParlSpeech V2 dataset and the parliament's multimedia archive.[23] The list of sampled debates, their duration, and number of batches is found in Table A1 in Appendix A, as well as details on the downloading, preprocessing, and batching of the recordings.

The first of the recordings is sampled uniformly at random. This recording is used as a human

[23] Figure A1 in Appendix A shows the population of debates visually. Audio recordings starting October 2010 are publicly available to download at `https://www.ft.dk/aktuelt/webtv/`. Recordings from 2000-2009 can also be retrieved from a collection of older sound recordings digitized by the Danish Royal Library (`https://dansklyd.statsbiblioteket.dk/samling/folketingsforhandlinger/`). These are not included in the population.

benchmark in validating the diarization stage and automated compilation of reference segments. To establish a human benchmark, I manually annotate the first 50 speeches in the recording (corresponding to the first hour of the debate) with timestamps using a collar of half a second.[24] Timestamps are manually assigned based on the units already outlined by the ParlSpeech V2 corpora where a single row corresponds to a single speech, as defined by each parliament's official plenary protocol.[25] This also assigns speaker information such as name and party to each speech. The average speech duration is 89.4 and 8.9 seconds with a standard deviation of 68.6 and 10.8 seconds for non-chair and chair speeches, respectively (see Table A2 in Appendix A).

The remaining 20 recordings are sampled using a weighted scheme with larger weights attached to recordings that contain more speakers found in the first sampled recording. The recordings are selected with a weighted sampling to ensure that a sufficient amount of speakers with reference segments are used to evaluate the performance of the cosine-based speaker inference. For this task, each sample is batched, diarized, and resegmented to construct a set of speech segments for each recording. Each segment is manually assigned a speaker label.

5.2 Diarization Analysis

I first evaluate the diarization stage, comparing the timestamps generated by the system with the manually annotated timestamps. To score this, I rely on the widely used Diarization Error Rate (DER) metric, which simultaneously measures the ability to detect speech segments (i.e., when segments start and end) and to distinguish between speakers. A lower DER indicates that the system commits fewer errors. See Appendix B for further details. Because the recording is diarized in batches, the DER is computed accordingly.

The results are reported in Table 2. The pre-trained diarization system from `pyannote.audio` (Bredin et al. 2020; Bredin and Laurent 2021) commits nearly no error with an average DER on 0.02 across the batches and a standard deviation of 0.01. This corresponds to around 1-1.5

[24]The collar is used to reduce human-induced errors and means that timestamps can be, for example, 00:10:00.000 or 00:10:00.500 but not 00:10:00.300 or 00:10:00.700, respectively.

[25]Timestamps are only assigned to speeches if a speech figures in both the recording and the transcript.

Batch	Duration	Correct	Total	False alarm	Missed detection	Confusion	**DER**
1	600	562.8	568.5	0.8	5.7	0.0	**0.01**
2	600	562.6	580.0	7.1	3.6	13.8	**0.04**
3	600	554.5	561.0	4.4	5.5	1.0	**0.02**
4	600	520.9	532.0	7.2	11.1	0.0	**0.03**
5	73	69.9	70.5	0.5	0.6	0.0	**0.02**
	Avg.	454.1	462.4	4.0	5.3	2.9	**0.02**
	Std.	215.5	219.8	3.3	3.8	6.1	**0.01**
	Min.	69.9	70.5	0.5	0.6	0.0	**0.01**
	Max.	562.8	580.0	7.2	11.1	13.8	**0.04**

Table 2: Diarization statistics using the off-the-shelf system from pyannote.audio. No collar is used in computing the DER. Units are in second except the DER which refers to the metric.

seconds of error per minute of speech. In comparison, in their validation of capturing emotion in multimodal data (MEXCA), Lükena et al. (2024) find an average DER of 0.20 in televised debates leading up to the Dutch general election in 2021, an error almost ten times larger. The magnitude of this difference is likely explained by the minimal noise and overlapped speech characterizing parliamentary debates compared to campaign debates. A closer look at the individual components in the DER shows that the error primarily arises because of failure to distinguish speech from non-speech (i.e., "missed detection'), and secondarily because non-speech is misclassified as speech (i.e., "false alarm"). Importantly, the system rarely confuses speakers (i.e., "confusion").

5.3 Measurement Analysis

The diarization analysis showed that the pipeline is able to retrieve speech segments automatically at a low error rate, and I now evaluate how this translates into the accuracy of downstream measurement. The low DER indicates that errors in downstream measurements might also be low. To evaluate this, I focus on the vocal pitch, which has been used as a dependent variable in regression models in applied research, proxying a legislator's issue commitments (e.g., Dietrich, Hayes, and O'Brien 2019; Rittmann 2023; Rask 2024a), partisan conflict (e.g., Rask and Hjorth 2024),

and voting intentions (e.g., Dietrich, Enos, and Sen 2019). I compute the vocal pitch at the level of each speech using manually annotated and automatically annotated timestamps, respectively, and calculate the absolute difference between the two. Lower values indicate greater similarity. I report measures using the entire speech segment ("All speech") and voiced speech, i.e., non-zero estimates ("Voiced speech"). See Appendix C for details about the computation of pitch and the validation metric.

I report the results in Table 3. The average absolute difference is diminishingly low for both evaluations with a difference of only 0.85 Hz for "Voiced speech" and 2.79 Hz for "All speech". For the voiced measure, this amounts to less than one percent of the average pitch for both men and women. A manual inspection shows that the pitch differences between automated and manual annotations arise primarily in shorter speech segments and typically in non-speech parts of the segments such as pauses or transitions between speakers.

	All speech	Voiced speech
Avg.	2.79	0.85
Std.	5.31	1.94
Min.	0.00	0.00
Max.	27.91	8.65

Table 3: Absolute difference in pitch estimates (unit in Hz).

Another way to validate the measure is to look at the alignment between automatically and manually annotated timestamps. I investigate this by looking at the correlation between the pitch estimates of each set of timestamps. A simple correlation analysis shows that the measures align completely. When I only consider speech segments ten seconds or longer, the correlation coefficient is $\rho = 0.999$, and when I consider all speech segments, the coefficient is $\rho = 0.993$. This is visually illustrated in Figure 4. The figure illustrates the near-perfect linear relationship with all points on the 45-degree angle line (i.e., a slope of one). As expected, this shows that the automatically generated speech segments can produce highly accurate downstream measures. At a

minimum, it shows that potential errors do not arise because of annotation errors.

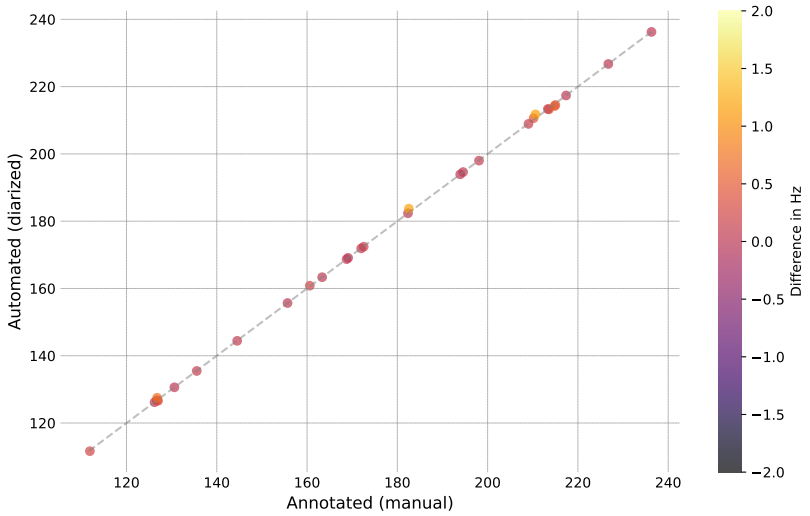


Figure 4: Relationship between pitch estimates computed with automated (i.e., diarized) and manually annotated timestamps. Speech-level pitch estimates are only shown for the "voiced speech" measure and only for speeches ten seconds or longer.

5.4 Compilation Analysis

The next evaluation pertains to the validity of the weakly supervised learning setup developed for the SR stage. The approach rests upon the joint availability of recordings, a corresponding pre-existing transcript, and the ability to link the former to the latter by matching ASR-generated transcripts (i.e., segment texts) to transcript texts.

I first construct segment texts using ASR on speech segments obtained through automated and manually annotated timestamps. As the diarization analysis showed virtually no difference see Table 2), I do not expect the type of timestamps to impact the performance of the weakly supervised setup substantially. The segment texts are generated using the "small" and "medium" ASR models from Whisper, respectively. The size of the language model (in terms of parameters) comes with a trade-off of accuracy vis-a-vis computational cost. A higher number of parameters generally yields a lower WER but also implies a significant speed reduction.[26]

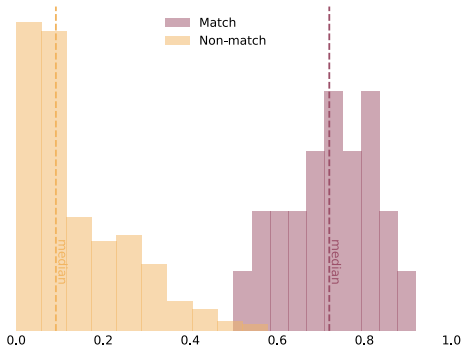
[26]The "small" model has 239 million parameters, and the "medium" model has thrice the number of parameters

After segment texts have been generated with each ASR model size for all speech segments, I manually label matched pairs of segment and transcript texts. If the approach works, those matched pairs should show up when the texts are linked using fuzzy string matching. Each segment text has one true transcript text match, and the remaining are non-matches.[27] To test this, I construct a similarity matrix containing the pairwise cosine similarity between each pair of segment and transcript texts. A higher similarity denotes that two texts are more similar in the sense that they share a higher portion of the same words by definition of the bag-of-words vectorization. Note that texts are generally not exactly similar for two reasons. First, while ASR models are generally very accurate, they still commit errors (Proksch et al. 2019; Tarr et al. 2022; Landesvatter et al. 2023). Second, if the pre-existing transcript is non-verbatim, a matched pair will not be exactly similar even if the ASR model is fully accurate.

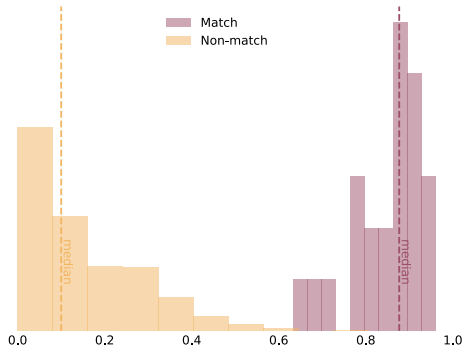
In Figure 5, I report the distribution of cosine similarity scores for matched and non-matched pairs of segment and transcript texts for manually and automated generated (i.e., diarized) timestamps and model sizes. The distributions show clear differences in the cosine scores for matched and non-matched pairs of segment and transcript texts independently of the type of timestamps and the model size. The median similarity differs substantially across matches and non-matches with only marginal overlap for the "small" model across the automated and manually annotated speech segments. For the "medium" model, there is no overlap. This shows that the similarity scores are driven more by the accuracy of the ASR models than by the type of timestamps, at least when the automated timestamps are as accurate as reported in the diarization analysis. Importantly, this error can be controlled by adjusting the threshold matching texts. For example, using $\tau = 0.60$ for the "small" model ensures perfect precision but with the cost of a lower recall. In Appendix D, I report the results using conventional classification metrics as a function of different thresholds. Based on the compilation analysis, I use $\tau = 0.70$ when compiling reference segments.

with 769. Compared to the "large" model, which has 1550 million parameters, the relative speeds of the "small" and "medium" models are ~6x and ~2x. This makes the "small" model around six times faster than the "medium" model. See https://github.com/openai/whisper for further information.

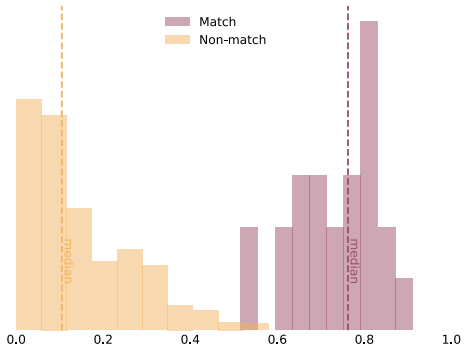
[27] This means that there is a total of N matches and a total of $N \times (M-1)$ non-matches.



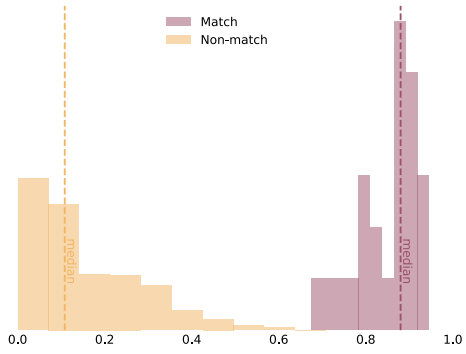
(a) Automated (“small”)



(b) Automated (“medium”)



(c) Manual (“small”)



(d) Manual (“medium”)

Figure 5: Cosine similarity scores between matched and non-matched pairs of segment and transcript texts using automatically annotated speech segments and manually annotated speech segments. The similarity scores are reported for both the “small” and the “medium” ASR model from Whisper. Only non-chair speeches are included in the validation.

5.5 Embedding Analysis

The compilation analysis showed that reference segments can be compiled automatically, making it possible to cast SR as a weakly supervised learning task. Before validating the speaker inference, I start by validating the usefulness of the embeddings used to encode the references for each speaker. A useful representation generates embeddings that are (1) similar for the same speaker and (2) dissimilar for different speakers. In other words, reference embeddings should show high intra-class and low inter-class similarity to accurately identify speakers in unseen speech segments.

I report both an informal and a formal test of these two properties. I start with the informal test

in Figure 6, visually presenting the reference embeddings after reducing the vectors to two dimensions using t-distributed Stochastic Neighbor Embedding (t-SNE) as implemented in `sklearn`. The reduced reference embeddings are shown colored by speakers. If the reference embeddings can distinguish between speakers, we should observe that embeddings uttered by the same speaker cluster. As evident from the figure, this is visually what happens. Embeddings uttered by the same speaker gravitate towards the same centroid. This informally supports the properties of the reference embeddings.

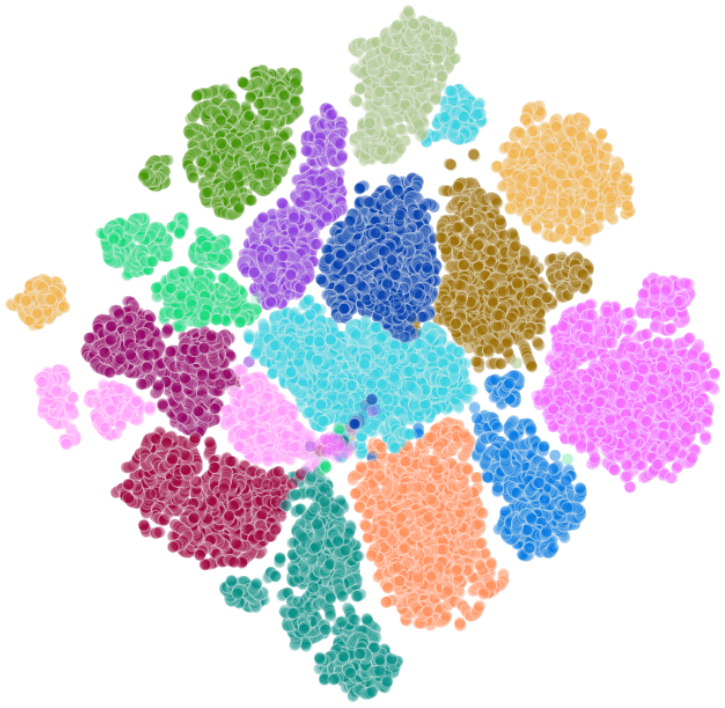


Figure 6: Reference embeddings (x-vectors) for speakers in `RES`. Embeddings are reduced to two dimensions with t-SNE using a learning rate of 200 and a perplexity of 50. For the visualization, the reference embeddings are computed using a sliding window with a duration of 1.6 seconds and a step of 0.0625. Each window has a 512-dimensional embedding.

I now turn to the formal test of intra-class and inter-class similarity. For this purpose, I compute

the pairwise cosine similarly between fixed-length reference embeddings stored in RES. Intra-class refers to the pairwise similarity within each speaker, and inter-class refers to the pairwise similarity between speakers. The result is reported in Table 4. The embeddings strongly encode both properties. The average intra-class similarity is 0.894 compared to an average inter-class similarity of 0.234. Importantly, the variance is also low with standard deviations of 0.038 and 0.051, respectively. This shows that reference embeddings appear useful in identifying and discriminating between speakers.

	Intra-class similarity	Inter-class similarity
Avg.	0.89	0.23
Std.	0.04	0.05
Min.	0.82	0.13
Max.	0.93	0.31

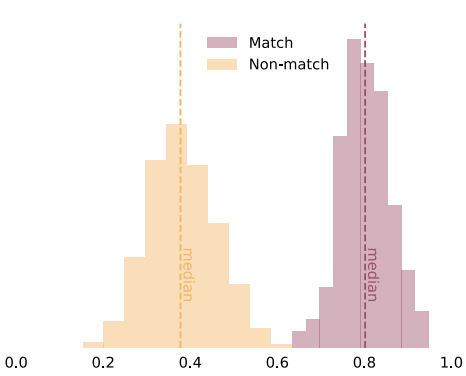
Table 4: Intra-class and inter-class similarity scores. For each speaker l, the intra-class similarity is the average pairwise cosine similarity between all pairs of reference embeddings (i.e. $\mathcal{R}^l$). The inter-class similarity is computed as the average pairwise cosine similarity between all reference embeddings for speaker l and all pairs of reference embeddings where $l \neq k$.

5.6 Speaker Analysis

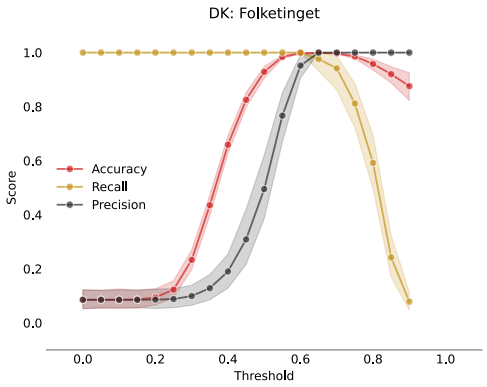
The last stage of the annotation pipeline is to infer the speakers of the speech segments identified with the diarization system using the reference embeddings compiled to the RES for the L target speakers. The embedding analysis suggests that reference embeddings prove useful in inferring speakers. To test, I investigate the extent to which speakers can be identified in hitherto unseen speech segments. The segments about to be annotated with speaker labels are processed similarly to those described for the diarization analysis (i.e., batching, diarization, and resegmentation). The classification follows the steps outlined and illustrated in Figure 3. The most similar reference embedding is computed for each segment and considered a candidate speaker. The candidate speaker is assigned as the actual speaker if the cosine similarity exceeds λ. I validate the inference

across different threshold values $\lambda \in \{0.0, \ldots, 0.9\}$, disentangling the trade-off and relationship between recall and precision.

The results are reported in Figure 7. I first consider the raw distribution of similarity scores for speakers in and not in RES. When a speaker has one or more reference segments, the similarity should be higher because the system should recognize the speaker more than speakers without references. This is what is observed in Figure 7a. The raw distribution of similarity scores shows that there is virtually no overlap between the scores for speakers with references (in RES) and those with no references (not in RES). I report corresponding summary statistics for the distribution in Appendix E. The right panel, Figure 7b, reports the classification results using different values of λ. This confirms the indications suggested by the similarity distribution. Speakers can simultaneously be classified with high precision and recall for certain thresholds. Considering the cost of false positives, using a threshold of $0.55 \leq \lambda \geq 0.75$ appears ideal to infer speaker identities from speech segments.



(a) Distribution of similarity scores for matches and non-matches (in RES / not in RES).



(b) Classification metrics as a function of cosine similarity thresholds $\lambda \in \{0.0, 0.1, \ldots, 0.9\}$.

Figure 7: Speaker identification analysis. Panel (a) shows the similarity distributions for speakers with and without reference segments. Panel (b) shows the classification metrics as a function of different λ thresholds.

6 Conclusion and Discussion

In an effort to advance the analysis of political speech recordings, this study introduces an annotation pipeline that automatically annotates audio recordings with human-level accuracy without the need for prior manually annotated data or any retraining. The pipeline relies entirely on pretrained DL-based models and a weakly supervised approach to speaker recognition based on fuzzy string matching. This automatism enables large-scale analysis of political speeches across languages and time. As political communication increasingly shifts from traditional text-based formats to audio and multimedia, there is a growing need for methods to handle vast amounts of audio data without relying on manual coding. While manual annotation remains valuable, the ability to analyze extensive audio datasets at scale is crucial for advancing the study of political speeches (Rheault and Borwein 2022a).

I validated the individual steps of the annotation pipeline using recordings of parliamentary debates from the Danish Parliament.[28] The overall results show that the pipeline is able to annotate audio recordings of political speech automatically at scale. It achieves a remarkably low DER with an average error of only around one second per minute of speech. Importantly, this leads to accurate downstream measurements of acoustic features. The speaker inference also commits virtually no error despite using reference segments compiled with a weakly supervised learning approach and not manually compiled references.

Before discussing the scope of the pipeline, a brief set of limitations deserves to be mentioned. First, the pipeline only concerns annotations related to the speech unit. The diarization system outputs segments and their corresponding timestamps at the level of each speech. However, other annotations carry substantive relevance for political science such as audience reactions to political debates like jeering (e.g., Ash, Krümmel, and Slapin 2024) or applause (e.g., Imre, Ecker, Meyer, and Müller 2023). This kind of annotation can be integrated into the pipeline in the future but

[28] I intend to validate the pipeline on recordings of parliamentary debates from other countries as well as different speech settings, e.g., televised campaign debates (Lükena et al. 2024), talk shows (Pinto 2024), or TV advertisements (Tarr et al. 2022).

requires an additional unit of analysis that works more at temporal rather than semantic units (e.g., Tumminia et al. 2021).

Second, the pipeline only concerns audio data but ignores the question of multimodal alignment (Rheault and Borwein 2019; Arnold and Küpfer 2024). The richness of human communication is jointly conveyed in multiple modalities, but political science tends to focus on one modality in isolation. As presented in this paper, the pipeline is primarily concerned with the audio signal of each speech. Yet, it is straightforward to align the audio with its corresponding text. The weakly supervised setup based on fuzzy string matching uses ASR to generate segment texts. This is done to compile reference segments but can also be used to align text and audio at the speech level.

Third, the pipeline currently ignores the video data available in most recordings of political speeches. Video data is the most recent advance in political science and offers yet another way to study political speeches (Nyhuis et al. 2021; Tarr et al. 2022; Girbau et al. 2024; Dietrich 2021; Dietrich and Sands 2023). This further speaks to the question of alignment and the multimodal nature of human speech. Adding video data to the pipeline is straightforward. It requires an additional layer of face detection, tracking, and classification to, for example, identify when a certain speaker appears in the video. The method developed by Girbau et al. (2024) might prove useful in combining the pipelines. This can be used to contextualize and understand the meaning of vocal cues even better, for example, by detecting whether a speaker reads out loud or looks at the audience.

Fourth, compiling reference segments for the SR stage of the pipeline relies entirely on the joint availability of audio recordings and pre-existing transcripts. On the surface, this seems like a strong limitation. However, this assumption is neither strong nor hypothetical. For instance, a large share of recordings of parliamentary debates has corresponding transcripts, found in, e.g., the ParlSpeech V2 (Rauh and Schwalbach 2020) or ParlaMint 2.0 datasets (Erjavec et al. 2023). Whenever a recording to be annotated has a corresponding pre-existing transcript, reference segments can be compiled in parallel with annotating the recording. If the recording to be annotated does not have a pre-existing transcript, reference segments can still be compiled with the weakly supervised setup.

Whenever a target speaker can be located in any recording-transcript pair, reference segments can be compiled and used in other recordings. This makes the approach highly flexible and makes it possible to compile reference segments in different settings.

These limitations aside, the scope of applications of the automated annotation pipeline is extensive due to its ability to compile large-scale datasets of political speech audio. The pipeline can be deployed to track how speaking time is distributed among different politicians, potentially revealing shifts in focus toward particular leaders or parties over time. Another promising application is to use the pipeline to explore disparities in speaking time based on gender or other demographic factors such as minority legislators. Furthermore, the pipeline can be used to identify patterns of interaction at a more nuanced level than in transcripts. This might allow us to uncover underlying networks among politicians or highlight specific debate formats and dynamics. More generally, it provides the basis for constructing datasets of text-audio speech for political science. The pipeline presented in this paper has been used to compile a text-audio dataset of all parliamentary speeches given in debates in the Danish Parliament from 2000-2022. This dataset has been used to study political representation, political power, and partisan conflict (Rask 2024a,b; Rask and Hjorth 2024). With audio-as-data becoming more prevalent in political science (Rheault and Borwein 2022b), and the still increasing availability of large-scale digitized audio archives, an automated annotation pipeline has the potential to unlock the promises of audio data.

References

Anastasopoulos, L Jason, Dhruvil Badani, Shiry Ginosar, and Jake Ryland Williams. 2024. "Visible home style". *Electoral Studies* 90 : 102794.

Arnold, Christian and Andreas Küpfer. 2024. "How alignment helps make the most of multimodal data". *arXiv preprint arXiv:2405.08454*.

Ash, Elliott, Johann Krümmel, and Jonathan B Slapin. 2024. "Gender and reactions to speeches in german parliamentary debates". *American Journal of Political Science*.

Back, Hanna, Marc Debus, and Jorge M Fernandes. 2021. "The politics of legislative debates". *The Politics of Legislative Debates*: 1.

Barari, Soubhik and Tyler Simko. 2023. "Localview, a database of public meetings for the study of local politics and policy-making in the united states". *Scientific Data* 10 (1): 135.

Bredin, Hervé. 2023. "pyannote.audio 2.1 speaker diarization pipeline: principle, benchmark, and recipe". In *Proc. INTERSPEECH 2023*.

Bredin, Hervé and Antoine Laurent. 2021, August. "End-to-end speaker segmentation for overlap-aware resegmentation". In *Proc. Interspeech 2021*, Brno, Czech Republic.

Bredin, Hervé, Ruiqing Yin, Juan Manuel Coria, Gregory Gelly, Pavel Korshunov, Marvin Lavechin, Diego Fustes, Hadrien Titeux, Wassim Bouaziz, and Marie-Philippe Gill. 2020, May. "pyannote.audio: neural building blocks for speaker diarization". In *ICASSP 2020, IEEE International Conference on Acoustics, Speech, and Signal Processing*, Barcelona, Spain.

Castanho Silva, Bruno, Danielle Pullan, and Jens Wäckerle. 2024. "Blending in or standing out? gendered political communication in 24 democracies". *American Journal of Political Science*.

Cochrane, Christopher, Ludovic Rheault, Jean-François Godbout, Tanya Whyte, Michael W-C

Wong, and Sophie Borwein. 2022. “The automatic analysis of emotion in political speech based on transcripts”. *Political Communication* 39 (1): 98–121.

Coria, Juan M., Hervé Bredin, Sahar Ghannay, and Sophie Rosset. 2020. “A Comparison of Metric Learning Loss Functions for End-To-End Speaker Verification”. In L. Espinosa-Anke, C. Martín-Vide, and I. Spasić (Eds.), *Statistical Language and Speech Processing*, pp. 137–148. Springer International Publishing.

Damann, Taylor J, Dean Knox, and Christopher Lucas. 2024. “A framework for studying causal effects of speech style: Application to u.s. presidential campaigns”.

de Slegte, Jef, Filip Van Droogenbroeck, Bram Spruyt, Sam Verboven, and Vincent Ginis. 2024. “The use of machine learning methods in political science: An in-depth literature review”. *Political Studies Review*: 14789299241265084.

Dietrich, Bryce J. 2021. “Using motion detection to measure social polarization in the us house of representatives”. *Political Analysis* 29 (2): 250–259.

Dietrich, Bryce J, Ryan D Enos, and Maya Sen. 2019. “Emotional arousal predicts voting on the us supreme court”. *Political Analysis* 27 (2): 237–243.

Dietrich, Bryce J, Matthew Hayes, and Diana Z O’Brien. 2019. “Pitch perfect: Vocal pitch and the emotional intensity of congressional speech”. *American Political Science Review* 113 (4): 941–962.

Dietrich, Bryce J and Melissa L Sands. 2023. “Seeing racial avoidance on new york city streets”. *Nature human behaviour* 7 (8): 1275–1281.

Erjavec, Tomaž, Maciej Ogrodniczuk, Petya Osenova, Nikola Ljubešić, Kiril Simov, Andrej Pančur, Michał Rudolf, Matyáš Kopp, Starkaður Barkarson, Steinþór Steingrímsson, et al. 2023. “The parlamint corpora of parliamentary proceedings”. *Language resources and evaluation* 57 (1): 415–448.

Furui, Sadaoki. 2005. "50 years of progress in speech and speaker recognition research". *ECTI Transactions on Computer and Information Technology (ECTI-CIT)* 1 (2): 64–74.

Girbau, Andreu, Tetsuro Kobayashi, Benjamin Renoust, Yusuke Matsui, and Shin'ichi Satoh. 2024. "Face detection, tracking, and classification from large-scale news archives for analysis of key political figures". *Political Analysis* 32 (2): 221–239.

Grimmer, Justin, Margaret E Roberts, and Brandon M Stewart. 2022. *Text as data: A new framework for machine learning and the social sciences*. Princeton University Press.

Herzog, Alexander and Kenneth Benoit. 2015. "The most unkindest cuts: speaker selection and expressed government dissent during economic crisis". *The Journal of Politics* 77 (4): 1157–1175.

Imre, Michael, Alejandro Ecker, Thomas M Meyer, and Wolfgang C Müller. 2023. "Coalition mood in european parliamentary democracies". *British Journal of Political Science* 53 (1): 104–121.

Joo, Jungseock, Erik P Bucy, and Claudia Seidel. 2019. "Automated coding of televised leader displays: Detecting nonverbal political behavior with computer vision and deep learning".

Kappos, Cybele. 2024. *The Way EU Make Me Feel: Measuring Anxiety in the Brexit Negotiations Using Text and Audio*. Ph. D. thesis, University of California, Los Angeles.

Karu, Martin and Tanel Alumäe. 2018. "Weakly supervised training of speaker identification models". *arXiv preprint arXiv:1806.08621*.

Kaufman, Aaron R and Aja Klevs. 2022. "Adaptive fuzzy string matching: How to merge datasets with only one (messy) identifying field". *Political Analysis* 30 (4): 590–596.

Knox, Dean and Christopher Lucas. 2021. "A dynamic model of speech for the social sciences". *American Political Science Review* 115 (2): 649–666.

Knox, Dean, Christopher Lucas, and Wendy K Tam Cho. 2022. "Testing causal theories with learned proxies". *Annual Review of Political Science* 25 (1): 419–441.

Kuhn, Harold W. 1955. "The hungarian method for the assignment problem". *Naval research logistics quarterly* 2 (1-2): 83–97.

Landesvatter, Camille, Jan Behnert, and Paul Cornelius Bauer. 2023. "Comparing speech-to-text algorithms for transcribing voice data from surveys".

Lükena, Malte, Kody Moodleya, Eva Viviania, Christian Pipalb, and Gijs Schumacherc. 2024. "Mexca–a simple and robust pipeline for capturing emotion expressions in faces, vocalization, and speech". *Working paper*.

Messeri, Lisa and MJ Crockett. 2024. "Artificial intelligence and illusions of understanding in scientific research". *Nature* 627 (8002): 49–58.

Neumann, Markus. 2019. "Hooked with phonetics: The strategic use of style-shifting in political rhetoric". In *Annual Meeting of the American Political Science Association. Washington, DC*.

Nyhuis, Dominic, Tobias Ringwald, Oliver Rittmann, Thomas Gschwend, and Rainer Stiefelhagen. 2021. "Automated video analysis for social science research 1". In *Handbook of Computational Social Science, Volume 2*, pp. 386–398. Routledge.

Park, Tae Jin, Naoyuki Kanda, Dimitrios Dimitriadis, Kyu J Han, Shinji Watanabe, and Shrikanth Narayanan. 2022. "A review of speaker diarization: Recent advances with deep learning". *Computer Speech & Language* 72 : 101317.

Peng, Zhiyuan, Xuanji He, Ke Ding, Tan Lee, and Guanglu Wan. 2022. "Unifying cosine and plda back-ends for speaker verification". *arXiv preprint arXiv:2204.10523*.

Peterson, Andrew and Arthur Spirling. 2018. "Classification accuracy as a substantive quantity of interest: Measuring polarization in westminster systems". *Political Analysis* 26 (1): 120–128.

Pinto, Gabriele. 2024. "Political talk show videos as data". *Available at SSRN*.

Plaquet, Alexis and Hervé Bredin. 2023. "Powerset multi-class cross entropy loss for neural speaker diarization". In *Proc. INTERSPEECH 2023*.

Proksch, Sven-Oliver, Will Lowe, Jens Wäckerle, and Stuart Soroka. 2019. "Multilingual sentiment analysis: A new approach to measuring conflict in legislative speeches". *Legislative Studies Quarterly* 44 (1): 97–131.

Proksch, Sven-Oliver, Christopher Wratil, and Jens Wäckerle. 2019. "Testing the validity of automatic speech recognition for political text analysis". *Political Analysis* 27 (3): 339–359.

Pruzansky, Sandra and Max V Mathews. 1964. "Talker-recognition procedure based on analysis of variance". *The Journal of the Acoustical Society of America* 36 (11): 2041–2047.

Radford, Alec, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2023. "Robust speech recognition via large-scale weak supervision". In *International conference on machine learning*, pp. 28492–28518. PMLR.

Rask, Mathias. 2024a. "Committed but constrained: Explaining why the descriptive-to-substantive representation link weakens over time". *Working paper*.

Rask, Mathias. 2024b. "When they go high, we go low: Rhetorical rewards of governing". *Working paper*.

Rask, Mathias and Frederik Hjorth. 2024. "Partisan conflict in nonverbal communication".

Rauh, Christian and Jan Schwalbach. 2020. "The ParlSpeech V2 data set: Full-text corpora of 6.3 million parliamentary speeches in the key legislative chambers of nine representative democracies".

Rheault, Ludovic and Sophie Borwein. 2019. "Multimodal techniques for the study of a ect in political videos". Technical report, Working Paper.

Rheault, Ludovic and Sophie Borwein. 2022a. *Audio as Data*. Edward Elgar Publishing.

Rheault, Ludovic and Sophie Borwein. 2022b. "Audio as data". In A. Ceron (Ed.), *Elgar Encyclopedia of Technology and Politics*, pp. 86–90. Edward Elgar Publishing.

Rittmann, Oliver. 2023. "Legislators' Emotional Engagement with Women's Issues: Gendered Patterns of Vocal Pitch in the German Bundestag". *Forthcoming in British Journal of Political Science*.

Rittmann, Oliver, Tobias Ringwaldy, and Dominic Nyhuis. 2020. "Pay attention to this! Explaining emphasis in legislative speech using automated video analysis in the US House of Representatives".

Silva, Bruno Castanho and Sven-Oliver Proksch. 2022. "Politicians unleashed? political communication on twitter and in parliament in western europe". *Political science research and methods* 10 (4): 776–792.

Slapin, Jonathan B and Sven-Oliver Proksch. 2008. "A scaling model for estimating time-series party positions from texts". *American Journal of Political Science* 52 (3): 705–722.

Snyder, David, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur. 2018. "X-vectors: Robust dnn embeddings for speaker recognition". In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 5329–5333. IEEE.

Tarr, Alexander, June Hwang, and Kosuke Imai. 2022. "Automated coding of political campaign advertisement videos: An empirical validation study". *Political Analysis*: 1–21.

Tiwari, Vibha. 2010. "Mfcc and its applications in speaker recognition". *International journal on emerging technologies* 1 (1): 19–22.

Torres, Michelle. 2024. "A framework for the unsupervised and semi-supervised analysis of visual frames". *Political Analysis* 32 (2): 199–220.

Torres, Michelle and Francisco Cantú. 2022. "Learning to see: Convolutional neural networks for the analysis of social science data". *Political Analysis* 30 (1): 113–131.

Tumminia, Jeffrey, Amanda Kuznecov, Sophia Tsilerides, Ilana Weinstein, Brian McFee, Michael Picheny, and Aaron R Kaufman. 2021. "Diarization of Legal Proceedings. Identifying and Transcribing Judicial Speech from Recorded Court Audio". *arXiv preprint arXiv:2104.01304*.

Variani, Ehsan, Xin Lei, Erik McDermott, Ignacio Lopez Moreno, and Javier Gonzalez-Dominguez. 2014. "Deep neural networks for small footprint text-dependent speaker verification". In *2014 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 4052–4056. IEEE.

Wang, Yu. 2024. "Leveraging large language models for fuzzy string matching in political science". *arXiv preprint arXiv:2403.18218*.

Williams, Nora Webb, Andreu Casas, and John D Wilkerson. 2020. *Images as data for social science research: An introduction to convolutional neural nets for image classification*. Cambridge University Press.

Online Appendix

Contents

A Validation Data: Sampling and Preprocessing

The population of debates using to sample recordings in the validation analysis is determined by the joint availability of pre-existing transcripts from the ParlSpeech V2 corpora (Rauh and Schwalbach 2020) and audio recordings from *Folketinget's* multimedia archive. This is shown in Figure A1. After sampling the recordings, each is downloaded from the parliament's multimedia archive (`https://www.ft.dk/aktuelt/webtv/`). Recordings are only available for download in video format (`.mp4`). After downloading each `.mp4` file, they are immediately converted to `.wav` files by discarding the video channel using the command-line tool `ffmpeg`. The audio recordings are then preprocessed to have a single channel (monaural compared to stereophonic sound) and a sampling rate of 16 kHz. This is also done using `ffmpeg`. Recordings are then batched using a duration of $d = 600$ seconds for each batch before being diarized and resegmented. After the output of the diarization system has been resegmented, only speech segments that last at least two seconds are kept for the analysis.

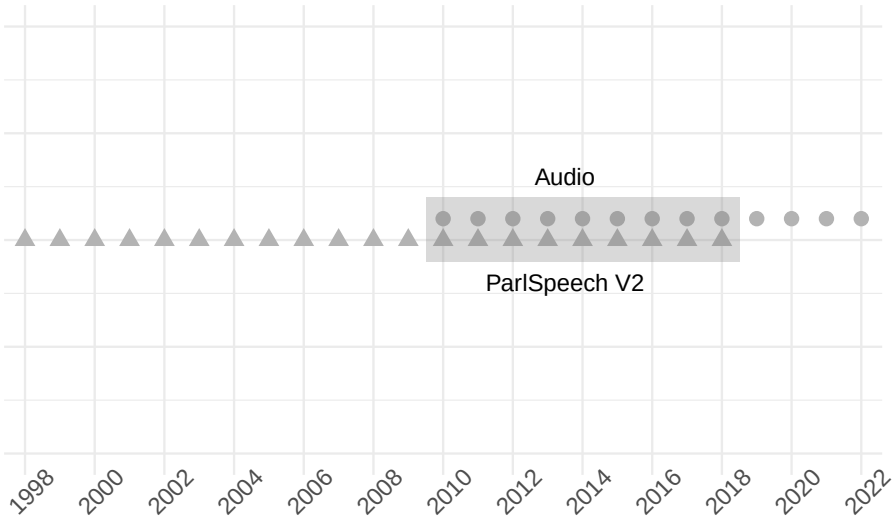


Figure A1: Joint availability of audio recordings from the Folketing's archive and pre-existing transcripts from ParlSpeech V2 (Rauh and Schwalbach 2020). Shared area corresponds to jointly available audio and pre-existing transcripts. This constitute the population of debates used to construct the human benchmark, excluding debates with few and a lot of speeches defined as the 5th and 95th percentile.

	debate	duration (s)	# batches	# speeches	# speakers
1	20171-97	31362.0	53	309	29
2	20161-94	32514.4	55	328	38
3	20151-97	40778.1	68	417	45
4	20151-30	11158.3	19	137	21
5	20151-90	27557.7	46	327	24
6	20131-90	34320.9	58	318	31
7	20181-37	20258.3	34	238	35
8	20161-42	12081.3	21	98	26
9	20161-5	11253.6	19	190	32
10	20151-32	43978.5	74	549	50
11	20161-20	18566.0	31	203	42
12	20161-41	12987.1	22	280	29
13	20171-56	24301.9	41	280	28
14	20161-95	33120.2	56	423	55
15	20171-10	38015.7	64	422	41
16	20161-12	32883.9	55	419	52
17	20141-89	21964.3	37	251	39
18	20121-83	14267.3	24	96	18
19	20161-91	9328.4	16	179	25
20	20131-33	17922.1	30	163	33

Table A1: Sampled debates: Duration, bathces, and the number of speeches and unique (non-chair) speakers.

Speaker	Avg.	Sd.	Min.	Max.	# Speeches	# Speakers
Non-chair	89.44	68.61	14.50	246.00	24	11
Chair	8.88	10.83	0.50	36.50	26	1

Table A2: Summary statistics for the speech duration and number of unique speakers for the 50 manually annotated speeches for the first sampled debate (20151-53).

B Diarization Error Rate

The Diarization Error Rate is the most commonly used metric to evaluate diarization systems and is defined as:

$$\texttt{DER} = \frac{\texttt{false alarm} + \texttt{missed detection} + \texttt{confusion}}{\texttt{total}}$$

The metric jointly captures a system's ability to retrieve speech segments, that is when a speech segment starts and ends, and to distinguish between different speakers.[1] The `false alarm` captures the duration of non-speech classified as speech analogous to a false positive or type I errors in hypothesis testing. Likewise, `missed detection` captures the duration of speech falsely classified as non-speech amounting to a false negative or a type II error. Lastly, `confusion` encodes the amount of speech assigned to a wrong speaker. The `total` parameter is the total amount of speech in the recording measured in seconds. By construction, lower values indicate a better performance with a lower bound of zero. Note that the measure seemingly has the look of a percentage, but the upper bound can exceed one in rare cases. The DER punish errors by duration but equally across speeches. In comparison, the Jaccard Error Rate (JER), based on the Jaccard similarity index measuring the similarity between two sets of segments, punish errors by each speaker's contribution but equally across speech duration.[2]

I compute the DER at the level of each batch and report the average DER and its standard deviation (SD) using no forgiveness collar. Forgiveness collar can accommodate uncertainty in the segment boundaries, for instance, imposed by the annotating procedure only assigning timestamps at half-second interval. In this case, this would suggest using a collar of half a second due to the annotation procedure used to construct the human benchmark. Hence, using zero collar in this case is an upper bound of the DER. The diarized and manually annotated speaker labels are matched within batches using the Hungarian algorithm in order to evaluate the confusion (Kuhn 1955).

[1] `pyannote.github.io/pyannote-metrics/reference.html`.

[2] `https://github.com/Picovoice/speaker-diarization-benchmark`.

C Pitch Measurement and Validation Metric

The measurement analysis is done by computing the speech-level vocal pitch using the Parselmouth library in `Python`, comparing estimates using the timestamps automatically annotated in the SD stage ($\hat{\rho}_i$) to manually annotated timestamps (ρ_i). The former can be considered an estimate of the latter. An unbiased estimate means that the two measures should be indistinguishable from each other. I capture this by calculating the absolute difference for each speech segment as:

$$\delta_i^{abs} = |\hat{\rho}_i - \rho_i| \tag{1}$$

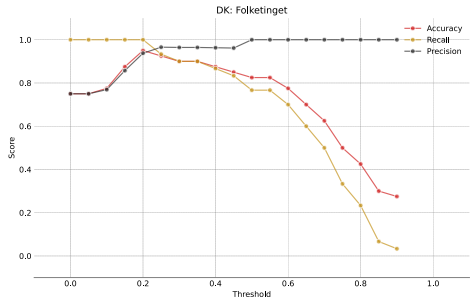
with lower values (lower bounded by zero) meaning that the timestamps yield identical speech-level measures of vocal pitch. Using the absolute difference means that over- and underestimates are treated equally.

I report the results using pitch estimates for the entire speech segment ("All speech") and for voiced speech only ("Voiced speech"). It is common practice to use only voiced speech when computing the pitch, but both measures are reported for transparency.

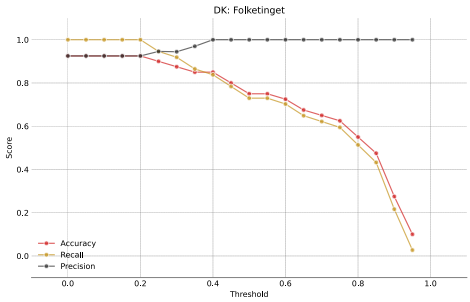
	Batch	Avg.	Std.	Min.	Max.
All speech	1	5.97	10.17	0.04	27.92
	2	3.58	3.70	0.01	8.85
	3	4.17	5.83	0.00	17.54
	4	6.53	11.43	0.00	39.33
	5	0.25	0.28	0.06	0.45
Voiced speech	1	2.12	3.67	0.00	8.65
	2	0.29	0.34	0.03	0.87
	3	0.92	1.80	0.00	6.40
	4	2.39	4.92	0.00	19.75
	5	0.17	0.07	0.12	0.22

Table C1: Absolute difference in pitch estimates (in Hertz) at the level of each batch.

D Fuzzy String Matching and Classification Report



(a) Automated (“small”)



(b) Automated (“medium”)

Figure D1: Evaluation scores as a function of different thresholds, $\tau = \{0.0, 0.1, \ldots, 0.9\}$. The texts include all speeches (both chair and non-chair).

E Speaker Inference and Similarity Scores

	Not in RES	In RES
Average	0.382	0.802
Standard deviation	0.077	0.056
Minimum	0.153	0.634
Maximum	0.633	0.948

Table E1: Summary statistics for similarity scores for speakers with (in RES) and without (not in RES) reference segments.