Degree Project in Computer Science and Engineering

Second cycle, 30 credits

Query-by-Example Audio Search using Acoustic Word Embeddings

Transforming wav2vec 2.0 Embeddings using Contrastive Learning

WILHELM ÖBERG

Query-by-Example Audio Search using Acoustic Word Embeddings

Transforming wav2vec 2.0 Embeddings using Contrastive Learning

WILHELM ÖBERG

Degree Programme in Computer Science and Engineering
Date: July 3, 2025

Supervisors: Jonas Beskow, Tomas Lundberg
Examiner: Olov Engwall
School of Electrical Engineering and Computer Science
Host company: Tonar Technologies AB
Swedish title: Query-by-Example Ljudsök med Akustiska Ordinbäddningar
Swedish subtitle: Omvandling av wav2vec 2.0-inbäddningar med Kontrastiv Inlärning

Abstract

The daily creation and consumption of audio and video content has grown dramatically, making large-scale speech analysis essential for trendspotting and market analysis. While transcription-based models are common, they are often costly and prone to errors, especially in low-resource languages. Query-by-Example (QbE) audio search offers an intuitive way to retrieve spoken content by using audio queries instead of text. In this thesis, we investigate how contrastive embedding learning can improve clustering of acoustic word embeddings, to filter out speaker characteristics and background noise, and to enhance QbE search performance. Starting from pre-trained wav2vec 2.0 embeddings, we train a projection network using triplet loss to better align similar word instances while increasing separation between dissimilar words. We evaluate this approach in two phases: (1) a clustering evaluation, which measures the embedding space structure before and after transformation, and (2) a QbE-focused information retrieval evaluation to quantify improvements in word retrieval accuracy. Our experiments compare a Swedish pre-trained model named VoxRex to a multilingual model XLS-R and conclude that the Swedish model shows high potential for QbE audio search. In contrast, the XLS-R needs further work before being viable. This work contributes partly to going completely transcription-free in audio search, which is an ambitious but attainable goal that could save both time and computational costs for businesses in the automatic speech recognition field, and also to understanding how contrastive learning can be used to construct a phonetically structured embedding space.

Keywords

Query-by-Example, Audio Search, Contrastive Learning, wav2vec 2.0, Triplet Loss, Word Embedding

Sammanfattning

Det dagliga skapandet och konsumtionen av ljud- och videoinnehåll har ökat dramatiskt, vilket gör storskalig talanalys avgörande för trendspaning och marknadsanalys. Transkriptionsbaserade modeller är normen i industin, men de är ofta kostsamma och kan sakna träffsäkerhet, särskilt när det gäller språk med begränsad tillgänglighet av ljuddata. Query-by-Exapmle ljudsökning erbjuder ett intuitivt sätt att söka efter talat innehåll genom att använda en inspelning av ett sökord istället för text. I den här uppsatsen undersöker vi hur kontrastiv inlärning kan förbättra klustring av akustiska ordinbäddningar, för att filtrera bort specifika talares egenskaper och bakgrundsljud för att förbättra QbE-sökningsprestanda. Med utgångspunkt i förtränade wav2vec 2.0-inbäddningar tränar vi ett projektionsnätverk med målfunktion baserad på triplettförlust för att bättre anpassa liknande ordsegment samtidigt som separationen mellan orelaterade ordsegment ökar. Vi utvärderar detta tillvägagångssätt i två steg: (1) en klustringsutvärdering, som mäter strukturen i inbäddningsrummet före och efter transformation och (2) en QbE-fokuserad utvärdering av informationssökning för att kvantifiera förbättringar i ordsökningsnoggrannhet. Våra experiment jämför en svensk förtränad modell (VoxRex) med en flerspråkig förtränad modell (XLS-R) och drar slutsatsen att den svenska modellen visar stor potential för QbE-ljudsökning, medan XLS-R behöver ytterligare arbete innan den kan användas i praktiken. Denna uppsats bidrar delvis till att gå mot att göra ljudsök fritt från kostsam transkribering vilket hade gynnat företag i taligenkänningsbranchen, samt bidrar med förståelse för hur kontrastiv inlärning kan användas för att skapa ett fonetiskt klustrat inbäddningsrum.

Nyckelord

Query-by-Example, Ljudsök, Kontrastiv Inlärning, wav2vec 2.0, Triplettförlust, Ordinbäddning

Acknowledgments

I want to express my gratitude to the entire team at Tonar for their support throughout this thesis. In particular, I am grateful to Tomas Lundberg for his supervision and guidance during the project. I would also like to thank my KTH supervisor, Jonas Beskow, for his support and our constructive discussions.

Stockholm, July 2025
Wilhelm Öberg

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Global internet access and the availability of smartphones have enabled large amounts of content to be published and consumed at all times. Instead of texting, we send a voice message. Instead of searching the web, we prompt our mobile voice assistant for the weather or make it set an appointment in our calendar for us. It is all enabled by automatic speech recognition (ASR), which converts speech to text. Audio Search [1] is a subfield of ASR focused on searching for keywords or sounds in an audio database. Keyword Spotting (KWS), Text-Based Spoken Term Detection (TB-STD), or Query-by-Example Spoken Term Detection (QbE-STD) are all audio search techniques based on searching a database with a query.

Social media, entertainment, and news platforms contribute massively to daily audio and video data creation, where podcasts and short-form content have seen striking growth figures. Audio search enables large-scale analysis of this content, enabling individuals and organizations to detect trends, get insights, and monitor different markets. A system that can perform well across all languages, handle speech from varied speakers, detect named entities, and be robust against change in audio quality is essential when harvesting these massive volumes of audio data.

Machine Learning and the ASR Community

Modern ASR is based on self-supervised learning [2] (SSL), a machine learning paradigm that leverages large amounts of unlabeled data to learn speech features. Organizations such as Facebook AI and OpenAI publish pre-trained transformer-based models that encode acoustic features into

meaningful embeddings that can be fine-tuned for specific downstream tasks. Facebook AI released its wav2vec 2.0 model [3], which is a framework for SSL. This model became famous for its then state-of-the-art performance, and for using raw audio as input.

KBLab is the data science section at the National Library of Sweden (KB). They provide Swedish datasets based on their extensive collections and have contributed several large pre-trained acoustic models. Specifically, their VoxRex model [4], pre-trained on Swedish public service radio data, outperformed the leading Swedish ASR models at its release. KB is democratizing Swedish audio collections and increasing the accessibility of accurate, large pre-trained models, which are difficult and expensive to train.

Large datasets like Google's FLEURS [5] and Mozilla's Common Voice [6] have accelerated ASR research and become an industry standard for building and benchmarking models.

One Model for All Tasks

One advantage of large pre-trained acoustic models is their ability to be fine-tuned for various downstream tasks. Some common tasks include: transcription [3], speech synthesis and cloning [7], query-by-example (QbE) spoken term detection (STD) [8], voice activity detection (VAD) [9], speaker diarization [10], and named entity recognition (NER) [11]. What all these downstream tasks have in common is that they are enabled by the latent speech features that these large models have learned to represent human speech. Feature extraction is an essential part of the signal processing pipeline, displayed in Figure 1.1. Depending on the constraints and requirements, different features are more or less suitable.

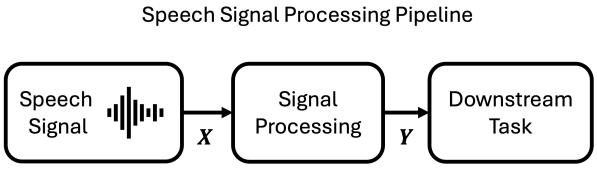

Figure 1.1: An example speech processing pipeline with a general downstream task based on processed signal Y.

A speech signal, denoted X, is usually represented as a vector where each element represents an amplitude x_t at time t. These digital signals are sampled at a fixed rate and quantized into discrete amplitude values, forming a discrete-time signal. In the signal processing step, different feature extraction methods

are applied to X, resulting in the feature vector Y. Depending on the chosen methods, which are influenced by the target task, the shape of Y is varied. Finally, Y is used to solve the desired downstream tasks.

1.2 Problem and Research Questions

Speech data is inherently complex and varied. Different speakers, accents, languages, and domain specific terms can make it difficult for transcription-based keyword search to achieve high performance. Transcribing text and manually reviewing transcription accuracy is not feasible for large volumes of audio data. Acoustic based search that detect similar utterances based on word-embeddings would remove the need for transcriptions entirely.

To approach this problem, we use wav2vec 2.0, a large pre-trained acoustic model to generate initial acoustic word embeddings (AWEs) that are used to train a lightweight multilayer perceptron. A contrastive learning objective is used to improve clustering of these embeddings. The use of contrastive learning transforms anchor and positive sample embeddings to lie closer in the embedding space while pushing away negative sample embeddings. By keeping the projected embeddings low-dimensional, vector search is nearly instant. Questions that will be addressed are:

- To what extent can the separability of wav2vec 2.0 speech embeddings be improved using embedding space transformation via contrastive learning? The degree of separation will be evaluated in high-dimensional space using clustering metrics such as Silhouette Score, Adjusted Rand Index (ARI), and Adjusted Mutual Information (AMI).
- Can transformed speech embeddings be effectively used for fast and accurate QbE audio search using a vector database? Retrieval quality will be assessed using binary classification-style metrics such as mean average precision (mAP), mean reciprocal rank (MRR), and recall@K.

Both research questions are addressed for two pre-trained wav2vec 2.0 models: A monolingual model pre-trained on Swedish speech and a cross-lingual model trained on 128 languages. Each model will be evaluated using the same embedding transformation architecture, training protocol, and similarly distributed datasets. The goal is to identify the strengths and limitations of each model's embedding space and retrieval performance in a head-to-head comparison.

1.3 Purpose

The purpose of this thesis is to develop a simple proof-of-concept embedding-based QbE audio search system for Tonar Technologies AB. The proposed system could be integrated into their existing speech analytics pipeline, adding acoustic-based search capabilities to complement their current arsenal. This approach bridges the gap between acoustic-based search and semantic, word-level search, without relying on transcriptions. In addition, this thesis contributes to the broader research on large pre-trained transformers. By developing a QbE audio search system using wav2vec 2.0 embedding transformations, the project provides insights into the structure and properties of the model's embedding space. Finally, this work compares the embedding spaces of a monolingual and a cross-lingual model to investigate how different pre-training strategies affect the usefulness of learned speech representations for audio search tasks.

1.4 Research Methodology

This work follows an experimental, engineering-driven research methodology. It involves developing a QbE audio search system, training an embedding projection network, and evaluating performance improvements using clustering and retrieval metrics for two pre-trained models. Using publicly available speech data and publishing this project's source code should facilitate reproducibility.

1.5 Delimitations

This thesis focuses on embedding space transformation and QbE retrieval. This project only considers word-level embeddings for the transformation of the embedding space and evaluation of QbE performance. It will not include unsupervised word-level segmentation of audio. Audio used for training and evaluating this model is extracted from publicly labeled datasets. Word-level segmentation of the datasets is performed with the provided ground truth audio transcriptions. For this model to be fully transcription-free, an unsupervised word-level segmentation algorithm needs to be added as a pre-processing step. This subject, which I highly encourage as future work, constitutes an entire thesis itself.

1.6 Structure of the Thesis

Chapter 2 presents the essential background theory required to understand all parts of the system and this thesis. Topics include: Embeddings as speech features, transformers, contrastive learning, QbE, vector search, clustering metrics, and classification metrics.

Chapter 3 covers the thesis methodology, a description of the contributed audio search system, the dataset creation process, and the evaluation plan.

Chapter 4 is divided into three sections. Firstly, a short account of the model training. Secondly, the clustering metrics and visualization of the embedding space transformation. Thirdly, QbE audio search system performance with retrieval metrics and example queries comparing before vs. after embedding transformation.

Chapter 5 finally states the thesis and experiment conclusions, answers the research questions, and suggests relevant topics for future work.

Chapter 2

Background

This chapter covers the required background knowledge related to concepts, methods, and models used in this project. Initially, we introduce embeddings, transformers and contrastive learning. Secondly, we move on to application focused background with a description of Query-by-Example and vector search. Then, the relevant evaluation metrics for both stages of the project, (1) embedding clustering and (2) binary classification for audio retrieval is presented.

2.1 Speech Embeddings and Representation Learning

2.1.1 From Input to Embedding

A speech embedding refers to a fixed size vector representation of a variable-length audio segment. The purpose of a speech embedding is to encode the meaning of an utterance. Embeddings in general can be used to represent objects at different abstraction levels. For example, in the natural language processing domain, word-level embeddings are created by encoding a short audio clip containing a single word utterance. However, there are cases where single character, phonetic unit embeddings or entire sentence embeddings can be relevant to encode, depending on the downstream task.

Representing the meaning of a word as a point in a high-dimensional embedding space offers several advantages. Consider the homonym *scale*, which could refer either to a piece of fish skin or to a weighing instrument. To capture the correct meaning, the context in which the word is used must be

taken into account. In representation learning [12], word embeddings typically incorporate some degree of contextual information.

However, when creating embeddings for word-level utterances spoken by different speakers, the goal shifts. Instead of capturing varying meanings, we aim to position utterances of the same word and phonetically similar words close together in the embedding space, despite variations in speaker characteristics such as accent, pitch, or speaking rate.

2.1.2 Properties of Good Embeddings for Speech

If we consider word-level embeddings for the audio search tasks, there are a number of good properties we would like these embeddings to have.

- When searching for a word in an audio database with a query, a perfect model should return all samples of the same word, irrespective of speaker-specific properties. The embedding should therefore not encode any information about pitch, accent, rate of speech, or background noise.
- We also look for natural clustering for the same or similar class utterances. This is simple to verify by querying the database for a keyword and returning the most similar samples. If they are of the same class or at least very similar, we can observe some natural clustering.
- Additionally, we want to see sparsity in the embedding space. When querying with a specific keyword, we want to have high similarity in positive samples and low similarity for other negative samples.

2.2 Self-Supervised and Contrastive Learning

SSL is widely used for learning latent representations and discovering structures in unlabeled data. It is often used in computer vision, video processing, and automatic speech recognition. SSL works by contrasting positive and negative samples to create generalized representations of objects that are highly correlated. By feeding these samples to the model, it is expected to learn distinguishable representations across classes.

Contrastive SSL is often used in a triplet network, as in Figure 2.1. The provided anchor and positive embedding are of the same class (word), and the negative embedding is from a different class. By applying transformations

to these embeddings to move positive samples closer together and negative samples further away, one can improve the class clustering and increase performance in downstream classification tasks.

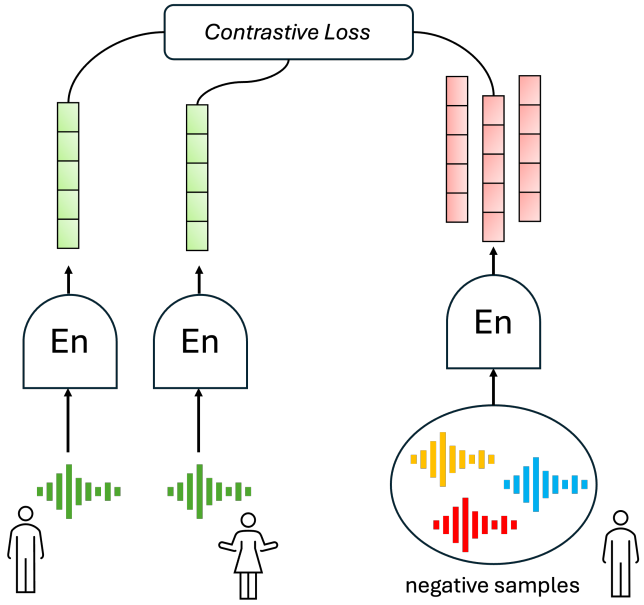

Figure 2.1: Contrastive SSL. Same word utterances (anchor and positive sample) from different speakers are paired with another word utterance (negative sample) to form a triplet.

2.3 The Original Transformer

The transformer is a specific neural network and the underlying architecture of most modern Large Language Models. It was introduced by Google in a famous paper called ”Attention is all you need” [13] to improve sequence-to-sequence modeling. It has, since its release, been used to create the famous Generative Pre-trained Transformer models. The main reason behind the massive adoption is the parallelizability during model training, which sharply reduced training time and complexity by eliminating the recurrency that its predecessors were limited by.

Transformers can be trained to solve different specific tasks, such as text translation or generation. The latter works by taking in a sequence of words (tokens) and predicting the most probable subsequent word based on the surrounding words (context), choosing the most probable token, and repeating

the process. The context window length affects how many tokens it will consider when predicting new tokens. A larger context will result in more coherent text generation.

The original transformer architecture contained some key components which made it superior compared to recurrent neural networks which dominated sequence to sequence modeling at the time. It follows an encoder-decoder structure as seen in Figure 2.2. It was originally intended for natural language processing tasks. However, its architecture has been used successfully across computer vision and audio applications. The core innovation of transformers is the self-attention mechanism, which lets the model assign importance to the context of its input when creating embeddings. Since it does not have recurrence, positional information is also encoded in each embedding of the input.

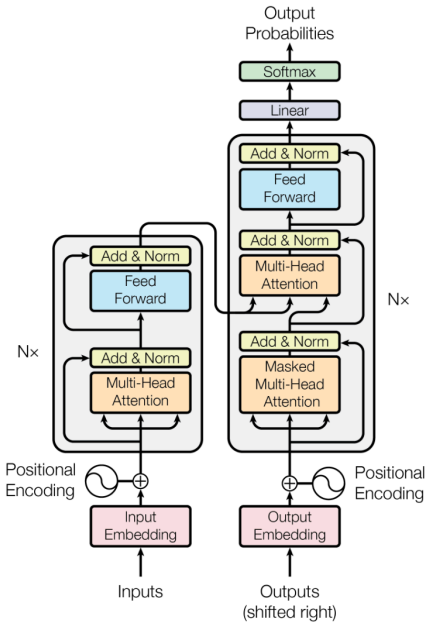

Figure 2.2: The Transformer - model architecture. With permission to reproduce from authors at Google [13].

Attention

The self-attention is what captures long-range dependencies of input sequences. It is based on three projections of every input token: Query (Q), Key (K), and Value (V). The purpose of these projections is to capture

different features of the sequence. In matrix form, the attention can be computed as in Equation 2.1.

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{2.1}$$

The attention score between two tokens is computed by taking the dot product of their Q and K projections. The result indicates if two tokens are relevant to each other. The softmax activation function is applied to obtain attention weights, determining which emphasis each token receives. The last step involves computing a weighted sum over attention weights and the value vector V. This is what gives each token its context for the entire sequence.

Transformers use multi-head attention (Equation 2.2), which enables learning different features simultaneously. Each head has its own Q, K, and V matrix. One can think of this as the multiple kernels in a CNN that capture different features by convolution. However, attention works globally on the entire sequence where a kernel operates on a token's near neighbors.

$$\begin{aligned} \text{MultiHead}(Q, K, V) &= \text{Concat}(\text{head}_1, ..., \text{head}_h)W^O \\ \text{where head}_i &= \text{Attention}(QW_i^O, QW_i^K, QW_i^V) \end{aligned} \tag{2.2}$$

In the encoder, the entire context is leveraged for each token to learn representations. However, in the decoder, a mask is applied to ensure that token prediction is only based on current and previous tokens. It works by setting the attention scores of future tokens to negative infinity before applying the softmax activation. This prevents the model from cheating by looking ahead during generation and enforces one-by-one word prediction.

2.4 Query-By-Example Spoken Term Detection

Query-by-Example Spoken Term Detection (QbE-STD) [1, §1.3] is an audio search technique where an audio database is searched with a spoken query. There is no transcription step in QbE-STD, meaning that matching is entirely based on acoustic feature similarity. A common technique that came before QbE-STD is keyword spotting (KWS). KWS is limited to detecting a certain set of predefined keywords from a transcribed audio clip. Similarly, Text-based STD systems use text search but with no constraint on what keywords they can

detect.

2.4.1 Embedding-Based QbE Systems

In an embedding-based QbE system, queries are represented as fixed-size vectors. Depending on the application, these representations are either word- or sentence-level embeddings. The key to achieving high-performance retrieval is the quality of embeddings and having a robust audio-to-embedding pipeline. In the case of word-level embeddings, longer audio clips must be segmented into words and encoded before similarity measurements can be computed. In Figure 2.3, an example QbE audio search pipeline is displayed.

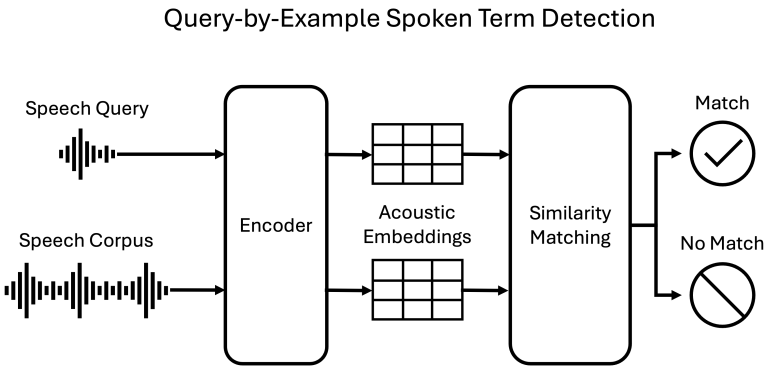

Figure 2.3: Query-by-Example Audio Search pipeline.

Vector Search by Cosine Similarity

Cosine similarity is a vector similarity measure that is defined as the cosine of the angle between the vectors according to Equation 2.3. Comparing vectors of fixed length with cosine similarity is very efficient and scales well with large databases. Typically, vector databases keep indexed embeddings and use measures like cosine similarity to facilitate either exact or approximate search.

$$\text{cosine similarity} = \frac{\mathbf{A} \cdot \mathbf{B}}{||\mathbf{A}||\,||\mathbf{B}||} \tag{2.3}$$

The cosine similarity always belongs to the closed set $[-1, 1]$. Where two proportional vectors have similarity 1, two orthogonal vectors have similarity 0, and two opposite vectors have similarity -1.

2.5 Evaluation of Embeddings and Retrieval

2.5.1 Clustering Evaluation Metrics

Word embeddings are fixed-size vectors that can be thought of as points in their high-dimensional embedding space. All word embedding samples belong to a class, which is the word they represent. To be able to analyze the structure of the embedding space, three metrics can be used: Silhouette score, adjusted rand index (ARI), and adjusted mutual information (AMI).

Silhouette Score

Silhouette score [14] is good for evaluating the geometry of the embedding space. It gives insight into how well the embedding space has natural structures, without using embedding labels. The score ranges from -1 to $+1$, where a high value a sample is well matched with its cluster. This metric is somewhat limited by the curse of dimensionality, meaning that this metric alone cannot be reliable when clusters are of varying sizes. It is calculated with any distance metric and assumes data has been clustered with k-means into k clusters.

Adjusted Rand Index

ARI [15] also leverages a k-means clustering of the embeddings and compares this clustering with the ground truth labels, corrected for chance by using an expected Rand index. Giving a precise measure of how well samples overlap. An ARI close to 0.0 indicates two random clusters, whereas a value of 1 shows identical clusters.

Adjusted Mutual Information

AMI [16] between two independent clusterings is a score that is useful for detecting agreement. This adjusted version of mutual information accounts for the fact that mutual information is higher for clusterings with a larger number of clusters. A value of 1.0 shows that the two clusters are identical, and a value of 0 indicates random clustering.

Dimensionality Reduction and Visualization

To illustrate embedding space transformations graphically, a projection technique called Uniform Manifold Approximation and Projection [17]

(UMAP) is useful. UMAP preserves both cluster layout and distance (global structure), is competitive with t-SNE for visualization quality, and is superior in run time performance. It is used as a general dimensionality reduction technique in machine learning and has no constraints on embedding size.

2.5.2 Query-by-Example Performance Metrics

The standard metrics for binary classification include accuracy, recall, false positive rate, and precision [1, §2.2]. For binary classification, a classification threshold will regulate these different metrics.

Threshold and Confusion Matrix

The final audio search system will be a binary classifier, deciding if a corpus contains a query or not. It is common to tune the model based on recall and precision targets.

	Actual Positive	Actual Negative
Predicted Positive	TP	FP
Predicted Negative	FN	TN

A high threshold $t = 1$ will bias "no match", and a low $t = 0$ will do the opposite.

Accuracy, Precision, Recall, and Related Metrics

$$\text{Accuracy} = \frac{\text{correct classifications}}{\text{total classifications}} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Recall (or TPR)} = \frac{\text{correctly classified actual positives}}{\text{all actual positives}} = \frac{TP}{TP + FN}$$

$$\text{FPR} = \frac{\text{incorrectly classified actual negatives}}{\text{all actual negatives}} = \frac{FP}{FP + TN}$$

$$\text{Precision} = \frac{\text{correctly classified actual positives}}{\text{everything classified as positive}} = \frac{TP}{TP + FP}$$

Mean Average Precision

When querying a vector database for a keyword, it will return samples in order of highest cosine similarity. mAP is commonly used in information retrieval to measure average precision@k with rank ordering in account. For each query in a set of queries, compute an average precision based on the ranked list of matches. The formula for a single query is:

$$\text{AP} = \frac{1}{N} \sum_{k=1}^{K} P(k) \cdot \text{rel}(k)$$

where:

- $P(k)$ is precision at rank k
- $\text{rel}(k) \in \{0, 1\}$: 1 if item at rank k is correct class
- N: number of relevant items in the dataset

Mean Reciprocal Rank

The MRR is the average of the reciprocal ranks of the top results from a set of queries. If a search returns a correct match for a given query, the rank for that query is 1, if the correct match was ranked second, it is ranked 0.5, etc.

Recall@K

Measuring recall@k means evaluating the recall of a keyword based on the top-k matches. A good model will retrieve all samples of the query word in order. Depending on the number of samples per keyword, recall@k can have some bias.

2.6 Related Work

2.6.1 Contrastive Learning for Imposing Structure in Latent Space

Dahlin [18] used contrastive learning with a custom loss function derived from Triplet loss to evaluate the possibility of attribute learning. Creating several variational auto-encoders that manipulated the latent space and imposed structure across both the visual and audio latent spaces. The focus of his

project was to perform data augmentation based on an embedded attribute. For example, for the MNIST digit images, his variational auto-encoder was able to rotate digits around an axis. Although he concluded that more work needs to be done to achieve high performance in his specific application, he showed that contrastive learning with Triplet loss is a viable option for embedding space structuring, which is exactly what this work will examine.

2.6.2 Unsupervised Pretraining Transfers Well across Languages

Transfer learning is a technique where knowledge from solving a specific task is used to improve the performance of a related task. In the case of phoneme recognition, it is intuitive that some languages have similar structures, and transfer learning should be useful when performing ASR. In a study [19], it was shown that a model pretrained in one language that predicted phonemes was usable across different languages for different ASR tasks. This indicates that learning acoustic units of a language should transfer well to downstream tasks such as query-by-example spoken term detection on word-level embeddings, which is interesting to examine.

2.6.3 Acoustic Word Embeddings Outperformed DTW in QbE

Settle et al. [20] have developed a QbE speech search system using word-level embeddings based on recurrent neural networks. Achieving substantial performance gains compared to traditional Dynamic Time Warping techniques and showing promising results for word-level embeddings in an audio search context.

The same authors [21] further developed and evaluated their QbE audio search system in a low- or zero-resource setting. Using the QUESST 2015 QbE task for benchmarking. This time, authors showed that an acoustic word embedding (AWE) based approach is much faster than traditional DTW-based search while outperforming previously published best models.

2.6.4 SSL Representations Useful in Various Downstream Tasks

In the article SUPERB [22], the authors establish a benchmark that is set to evaluate the current state of self-supervised learning models in different

downstream audio processing tasks. The framework aims to explore the power of representation learning from different pre-trained general speech models. The idea was to use embeddings from frozen models and train small, specialized networks in solving different audio-related downstream tasks such as: Phoneme recognition, keyword spotting, speaker identification, query-by-example, intent classification, speaker diarization, emotion recognition, and slot filling. So, a wide variety of downstream tasks were attempted using frozen embeddings from some pre-trained SSL-based models. Among the evaluated models, wav2vec 2.0 performed at the top level in keyword spotting and speaker diarization. The authors conclude that speech representations learned by wav2vec 2.0 are powerful and that training quality ASR systems has become easier than before.

Chapter 3

Method

This chapter details the proposed audio search system. It also covers the essential step of data creation and data engineering. Furthermore, we cover the evaluation plan for the two tasks: (1) Embedding space transformation, and (2) vector search using a QbE set-up.

3.1 Engineering Research Approach

This work makes use of large pre-trained transformer-based models for creating acoustic word embeddings (AWE). These embeddings are later projected to improve clustering to facilitate audio search.

When training a neural network, an iterative approach is common. The distribution of training and testing data needs to be optimized to obtain a model that generalizes well. This is done by repeated experimentation with varying setups. A QbE system is evaluated using metrics derived from accuracy, precision, and recall. This means that a quantitative evaluation method is applied in these experiments. This upcoming method section aims to give a clear picture of the research process, to facilitate reproducibility, and to let future work learn from and improve on this thesis.

3.2 Embedding Extraction from Pre-trained Models

3.2.1 wav2vec 2.0

wav2vec 2.0 [3] is a framework for SSL and is capable of learning acoustic representations from raw speech audio. It uses masking of the input in the latent space and solves a contrastive task. It can be used for a variety of different downstream tasks [22] and is a strong candidate for a QbE audio search system due to its ability to create powerful and expressive acoustic embeddings. The model consists of three stages: (1) Feature encoding by convolutional neural networks, (2) contextualized representations using transformers, and (3) a quantization module. See Figure 3.1 for a detailed illustration.

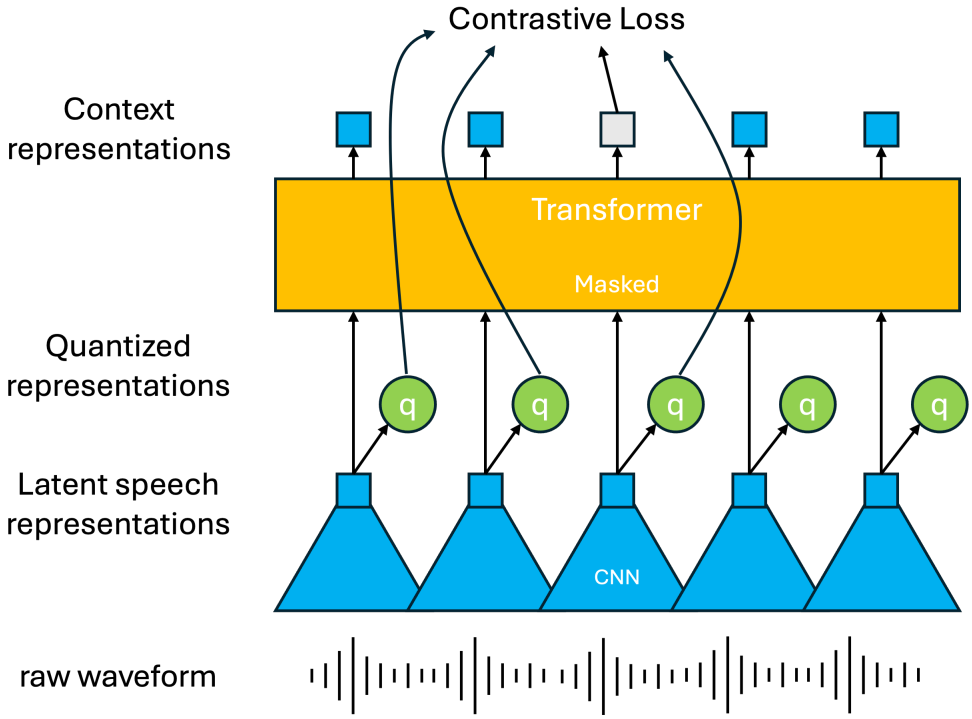

Figure 3.1: Illustration of the wav2vec 2.0 framework that learns context and speech units simultaneously.

When training this model, thousands of hours of unlabeled speech are required. This process is both time-consuming and expensive. Facebook AI

has released both mono- and cross-lingual versions of wav2vec 2.0.

When an audio file is fed to this model, high-dimensional acoustic embeddings are created for each time frame. Each frame in the embedded vector is approximately 20 ms and consists of 1024 values.

3.2.2 VoxRex: A Swedish wav2vec 2.0 Model

KBLab has leveraged its extensive audiovisual collections to train and release a Swedish wav2vec 2.0 model [4]. This model is trained on a Swedish speech corpus called P4. It consists mainly of local public radio, but also podcasts and audiobook data. P4 consists of 25 regional radio stations, and data from the last twenty years were included. Having radio data from different regions has resulted in diverse speech data, which is beneficial for democratizing the model. VoxRex follows the same architecture as the original wav2vec 2.0 large model. It has 300 million parameters and was trained on 11,100 hours of speech for 400,000 updates, which took approximately 21 days. After benchmarking, it was concluded that it outperformed the state-of-the-art mono- and cross-lingual models in transcription accuracy.

3.2.3 XLS-R: A Cross-Lingual wav2vec 2.0 Model

Researchers from Meta AI, Google AI, Outreach, and Hugging Face published their cross-lingual pretrained wav2vec 2.0 model XLS-R [23], used for speech representation learning. It was trained on nearly half a million hours of publicly available speech audio in 128 languages. It was found to perform equally good in translating English speech into other languages compared to English-only trained models. This model is specifically designed for representation learning tasks and was evaluated across three different downstream tasks: Automatic speech translation, automatic speech recognition (ASR), and language identification. Concluding that a cross-lingual model with sufficient capacity can perform as well as monolingually pretrained models.

3.3 Dataset Engineering

3.3.1 Google FLEURS Dataset

The FLEURS dataset [5] contains speech data in 102 languages, with each language contributing approximately 12 hours of audio with transcriptions.

The recordings vary in audio quality, with differences in background noise and microphone characteristics. Speech samples are read by a diverse set of speakers and are segmented at the sentence level, providing a broad representation of real-world variability. This dataset was chosen because of its high quality and feasible size, enabling experiments to run locally on device.

For this thesis, various subsets of FLEURS are used for both training and evaluation. Because this work focuses on acoustic word embeddings, the first step in preparing the data involved word-level segmentation, dividing the sentence-level audio into isolated word utterances.

The study specifically uses four languages: Swedish, Danish, Norwegian, and Finnish. These languages were chosen to cover both closely related Nordic languages and to introduce cross-lingual variation, enabling the analysis of how language similarity and phonetic differences affect the learned embeddings.

3.3.2 Forced Alignment and Word Segmentation

WhisperX [24] is an open-source model based on OpenAI's Whisper [25] model. It does transcription and forced alignment in a more scalable manner compared to the original Whisper model. Forced alignment is the task of segmenting words by predicting word-level time stamps based on an audio clip and associated transcription. It was used, in this work, as a supervised word segmentation tool to segment the above-mentioned language subsets of the FLEURS dataset. WhisperX makes use of publicly available wav2vec 2.0 models for the alignment backend, and the user is free to use whichever suits them the best. In this work, we proceeded to use Swedish VoxRex for alignment in Swedish, Danish, and Finnish, since the latter two do not have a pretrained version of wav2vec 2.0 in their respective language. For Norwegian, a pre-trained model released by NbAiLab, the AI-lab of the National Library of Norway, was used.

WhisperX Limitations

The lack of language specific pretrained models for Danish and Finnish led to some word segments being of inadequate quality for inclusion in the final experiment data. Affected words were mostly short and with few letters. These short words constitute the most common words and would also have been heavily overrepresented in the data. To counter this, specific filtering was applied to narrow down the list of eligible samples after segmentation.

3.3.3 Filtering and Splitting the Datasets

After word segmentation, the next step included determining the training, testing, and validation distributions. Because of the heavy word imbalance and poor quality of some words, sample filtering was implemented. For a sample to be considered for the dataset, it has to fulfill these requirements:

- Minimum samples per word: 2
- Maximum samples per word: 15
- Minimum sample duration: 0.4 seconds
- Minimum word length: 4

This basic filtering was necessary to extract mid to high-quality samples.

Training, Testing, and Validation Splits

There were 10294 word segment samples for a total of 3000 unique words included in the dataset. In Table 3.1, the dataset splits for the Swedish experiment are displayed.

Table 3.1: Word distribution and sample count for the Swedish dataset.

Split	**Unique Words**	**Total Samples**
Train	1800	6229
Validation	600	2035
Test	600	2030
Total	3000	10294

In Table 3.2, splits for the multilingual datasets are displayed. A total of 8873 samples for 2924 unique words were selected. All samples followed the filtering requirements outlined above.

External Validity Consideration

All experiments were conducted with designated training and test datasets. To aim for external validity of all models, test datasets contained words unseen during training. This helped during evaluations to indicate whether embedding transformations were correctly learned or if the model had overfitted.

Table 3.2: Word distribution and sample count across languages for the Multilingual dataset.

Split	Danish	Finnish	Norwegian	Swedish	Unique Words	Total Samples
Train	440	462	449	446	1754	5233
Validation	162	143	145	150	584	1857
Test	148	145	156	154	586	1783
Total	—-	—-	—-	—-	2924	8873

3.3.4 Triplet Dataset Construction

Contrastive Learning by triplet loss requires a special structure of the training data. The training objective is to minimize distance between anchor and positive pair, while increasing separation to the negative sample.

Compute Embeddings

At this stage, audio segments were encoded into word embeddings. This was done by feeding each segmented audio sample to the wav2vec 2.0 encoder. This was done for both VoxRex and XLS-R to create embeddings for their respective evaluation.

Mean Pooling

A crucial decision that significantly impacted this work is whether the embeddings were to be kept as a time series of embedding vectors or if mean pooling along the time axis should have been used to create a single embedding vector to represent each word. In this work, the latter was chosen for simplicity of storage and search. Having embedded matrices of different dimensions depending on the duration of an utterance would have added complexity to the embedding transformation network. The mean pooling was used as a type of downsampling to create embedding vectors of consistent dimension. The high dimensionality of the embedding, 1024D, was deemed sufficient to encode relevant features on a word level.

Random vs. Hard Triplet Sampling

Before constructing the triplet dataset, there were two strategies to consider. Either random sampling of anchor, positive, and negative samples could be used, which works well in most cases. If the model was having difficulty

separating clusters, hard triplets were implemented. A hard triplet is when the negative sample is chosen based on high cosine similarity to the anchor.

Measuring embedding similarity is simple when using cosine similarity. Finding hard triplets work by (1) randomly sampling an anchor word, (2) selecting a random positive sample among anchor class samples, and finally (3) selecting a negative sample with medium to high similarity to the anchor.

We evaluated the effects of both random and hard triplets in this thesis. A discussion on hard triplet implications can be found in section 4.4.

3.4 Embedding Transformation Network

The purpose of the embedding transformation network is to learn a projection that will increase class separability and restructure the embedding space so that phonetically similar words lie closer. Focusing the embeddings to encode phonetic content, not speaker-specific characteristics.

3.4.1 Architecture

This embedding transformation network has a shallow structure and was fast to train using contrastive learning. It was trained to transform 1024-dimensional wav2vec 2.0 embeddings into a 256-dimensional space optimized for vector search. It was implemented as a simple two-layer neural network $f : \mathbb{R}^{1024} \rightarrow \mathbb{R}^{256}$ with ReLU activation and L2-normalization.

The transformation network is defined as:

$$f(\mathbf{x}) = \frac{g(\mathbf{x})}{||g(\mathbf{x})||_2}$$

where:

$$g(\mathbf{x}) = \mathbf{W}_2 \cdot \text{ReLU}(\mathbf{W}_1 \cdot \mathbf{x} + \mathbf{b}_1) + \mathbf{b}_2$$

- $\mathbf{x} \in \mathbb{R}^{1024}$: the input embedding
- $\mathbf{W}_1 \in \mathbb{R}^{512\times1024}, \mathbf{b}_1 \in \mathbb{R}^{512}$: weights and bias for the hidden layer
- $\mathbf{W}_2 \in \mathbb{R}^{256\times512}, \mathbf{b}_2 \in \mathbb{R}^{256}$: weights and bias for the output layer
- $\text{ReLU}(z) = \max(0, z)$: the rectified linear unit activation
- $||g(\mathbf{x})||_2$: the L2-norm applied across the feature dimension

The final output $f(\mathbf{x}) \in \mathbb{R}^{256}$ lies on the unit hypersphere, allowing cosine similarity to be directly computed using the dot product for fast and simple retrieval.

3.4.2 Training

Objective and Triplets

The network was trained using triplet margin loss, where each training sample was a triplet $(\mathbf{x}_a, \mathbf{x}_p, \mathbf{x}_n)$ consisting of mean-pooled wav2vec 2.0 embeddings.

The loss function is defined as:

$$\mathcal{L}_{triplet} = \max(0, d(f(\mathbf{x}_a), f(\mathbf{x}_p)) - d(f(\mathbf{x}_a), f(\mathbf{x}_n)) + \alpha)$$

where d is the pairwise distance $d(x_i, y_i) = ||\mathbf{x}_i - \mathbf{y}_i||_2$, and α is a fixed margin that defines how far apart the positive and negative sample should be in the embedding space relative to the anchor. A small margin allows tighter, potentially overlapping clustering, and a large margin encourages better separation but can make training difficult or unstable.

Procedure, Hardware and Hyperparameters

The network was trained using the Adam optimizer with a learning rate of 1×10^{-4} and a batch size of 2048. The training loop ran for 1500 epochs and uses a validation set to compute silhouette scores every 20 epochs. The model at the epoch with the best validation score was saved. Triplets were sampled randomly across epochs, and 40 % of triplets were specifically created to have semi-hard negatives. This meant negatives had a cosine similarity to the anchor sample of 0.5 to 0.8.

In the forward pass, triplets were fed through the network, creating 256-dimensional projections which were L2-normalized. The triplet margin loss function used an annealing margin schedule starting from $\alpha = 0.55$ and ending at $\alpha = 0.8$ over the entire training. Gradients were backpropagated through the network, and the model weights were updated via Adam optimization. Each epoch compounds the total loss, which is displayed during training. Training times are usually very short for this shallow network.

Models were trained on an NVIDIA GeForce RTX 2070 SUPER GPU with 8 GB video memory and CUDA support.

3.5 Evaluation Plan

Two different embedding models were compared.

- A Swedish model: KBLab/wav2vec2-voxrex-swedish [4]
- A Multilingual model: facebook/wav2vec2-xls-r-300m [23]

Each model had their own embedding projection network trained on separate datasets. The Swedish model was trained on a Swedish triplet dataset, and the multilingual model was trained on a multilingual triplet dataset. These models were evaluated regarding embedding space structure and QbE vector search.

3.5.1 Phase 1 - Embedding Space Evaluation

The goal was to determine whether transformed embeddings exhibit improved class separability compared to original embeddings. This phase focused on the structure and geometry of the high-dimensional embedding space.

Evaluation Procedure

- Computed clustering evaluation metrics: Silhouette Score, Adjusted Rand Index, and Adjusted Mutual Information.
- Visualized embeddings using UMAP to project original and transformed embeddings to 2D.

3.5.2 Phase 2 - Query-by-Example using Vector Search

The goal was to evaluate the usefulness of embeddings in real retrieval tasks.

Evaluation Procedure

- The test set was used to create two FAISS databases [26]. One for the original and one for the transformed embeddings.
- For each query, the system retrieved the top k nearest neighbors using cosine similarity.
- Retrieval quality was measured using binary classification metrics mAP, MRR, and Recall@K. Some qualitative query examples for discussions were also presented.

Chapter 4

Results and Discussion

In this section, we present model training and experimental evaluation results. Firstly, in section 4.1, the different models are introduced with the corresponding training datasets. In section 4.2, the embedding space structure is evaluated and visualized for original and transformed embeddings. Finally, in section 4.3, the QbE evaluation provides information retrieval metrics to illustrate performance gains in the downstream task. Some example queries are brought up to facilitate qualitative analysis of the embedding transformations.

4.1 Model Training

Table 4.1 contains model training metrics. The Swedish model was trained for 3 minutes and 43 seconds using triplets constructed from the Swedish training split dataset Table 3.1. Training parameters were according to Section 3.4.2. The multilingual model X-Hard was trained for 3 minute and 59 seconds on a triplet dataset constructed according to Section 3.3.4, based on a training split of the multilingual dataset Table 3.2.

Table 4.1: Training summary for the two model configurations.

Model	**Emb. Model**	**Dataset**	**Time (m:s)**
SWE-Hard	VoxRex	Swedish	03:43
X-Hard	XLS-R	Multilingual	03:59

The Swedish model was trained for 1500 epochs and saw a sharp initial drop in loss, see Figure 4.1. The validation silhouette score peaked at approximately 0.20 around epoch 1300. Annealing during training meant

progressively increasing the difficulty of class separation by increasing the margin, and using harder triplets with higher cosine similarity over time.

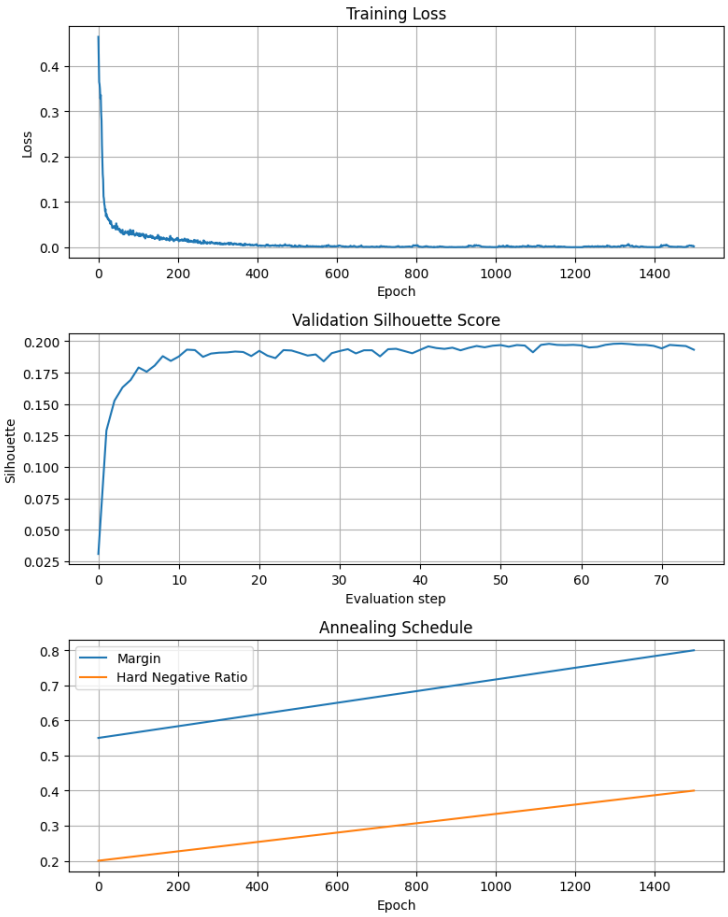

Figure 4.1: Training loss, validation silhouette score every 20 epochs, and annealing schedule for the Swedish model SWE-Hard.

In Figure 4.2, the X-Hard model training is displayed. It was trained for 1500 epochs and saw a sharp initial loss. Validation score was negative and peaked at -0.1. This could be due to the embedding space structure. It used the same annealing schedule as the Swedish model. The best model checkpoint for X-Hard was at epoch 320.

The best model checkpoints for both SWE-Hard and X-Hard were selected for the full evaluation plan, starting with Phase 1: Embedding space evaluation. And moving on to Phase 2: QbE evaluation.

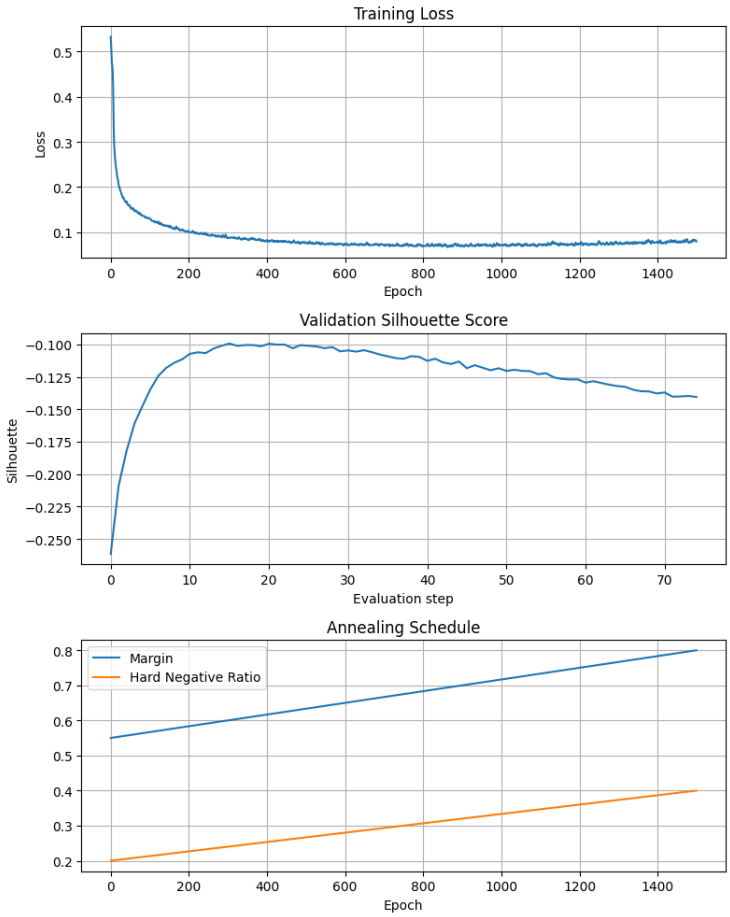

Figure 4.2: Training loss, validation silhouette score every 20 epochs, and annealing schedule fir the Multilingual model X-Hard.

4.2 Phase 1: Embedding Space Evaluation

4.2.1 Clustering Evaluation

Here we report the clustering of word samples in the embedding space before and after transformations. Table 4.2 shows the clustering scores: Silhouette score, ARI, and AMI. The Swedish-only model achieved the high scores with a silhouette score of 0.2399, meaning that classes have separated quite well. It attained an ARI of 0.5945 and an AMI of 0.6885, meaning that a k-means fitting of the projected sample embeddings is aligning well with the true sample clusters in the Swedish test dataset.

The multilingual model X-Hard achieves a silhouette score of 0.0925,

Table 4.2: Clustering evaluation metrics (Silhouette, ARI, NMI) for original and transformed embeddings across all model variants.

Model	Dataset	Embedding	Silhouette	ARI	AMI
XLS-R	Test-X	Original	0.040	0.046	0.090
X-Hard	Test-X	Transformed	0.093	0.234	0.334
VoxRex	Test-SWE	Original	0.084	0.164	0.276
SWE-Hard	Test-SWE	Transformed	**0.240**	**0.595**	**0.689**

which is far from SWE-Hard. However, it improved its silhouette score by a factor of 2.31, which is comparable to the Swedish model that had an improvement ratio of 2.84. This shows the importance of the starting point of the embedding space structure. If embeddings are well structured from the start, a simple network with minimal training can improve structures significantly. If the embedding structure is having difficulty discretizing samples from the start, a large improvement ratio will not be as impactful.

4.2.2 Word Cluster Visualization with UMAP

The second stage of phase 1 evaluations includes lower-dimensional visualization to get qualitative insight into the embedding space restructuring effects of the projection network.

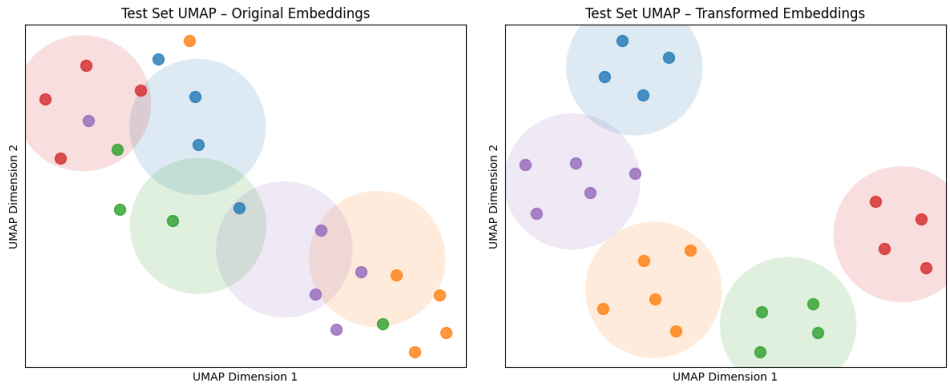

Figure 4.3: SWE-Hard: UMAP projections of original (left) and transformed (right) embeddings for the Swedish models. Color indicates word classes: 'talare', 'slutändan', 'aggressiva', 'jämföra', 'buffalo'.

In Figure 4.3, two UMAP visualizations of embeddings before and after transformation by the Swedish model are displayed. The words are randomly sampled from the test set and are unseen during training. Before transforming

embeddings using the network, the sample embeddings are not structured in word clusters. This illustrates the silhouette score, ARI, and AMI attained in stage 1 of phase 1 evaluations. The embedding space has now become ordered in word clusters.

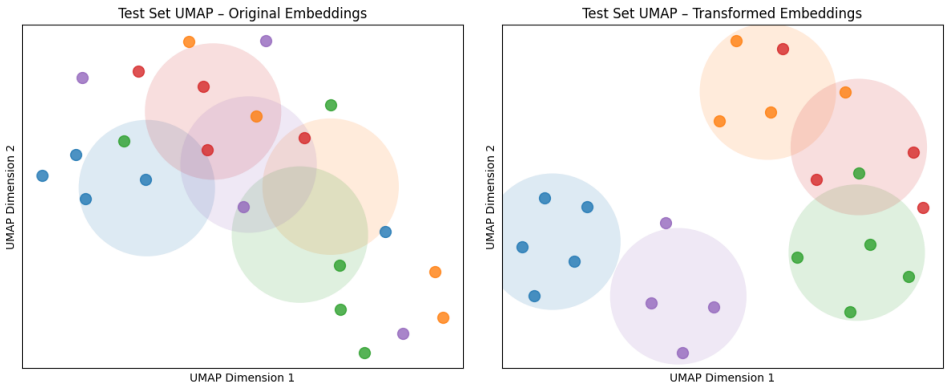

Figure 4.4: X-Hard: UMAP projections of original and transformed embeddings for XLS-R based multilingual model. Color indicates word classes: 'alkoi', 'redigera', 'efterfølgende', 'nærmest', 'parallelle'.

In Figure 4.4, UMAP visualizations of the original XLS-R embeddings and X-Hard transformed embeddings are displayed. Similarly to the Swedish model, original embeddings are randomly distributed across the embedding space. Transformed embeddings have aligned in word clusters, but not as closely as in the Swedish model.

4.3 Phase 2: QbE Retrieval Evaluation

In Phase 2 of the evaluation, the application capabilities of the newly trained models are evaluated. In particular, a Query-by-Example vector search system is implemented and evaluated using binary classification and information retrieval metrics.

4.3.1 Vector Search Evaluation

In Table 4.3, the information retrieval metrics mAP, MRR, and Recall@20 are reported for embedding search of both original VoxRex-based and transformed SWE-Hard embeddings. Initially, the model is somewhat aligned already with a mean average precision of 0.440, a mean reciprocal rank of 0.468, and a Recall@20 of 0.6. Results for transformed embeddings have significantly

increased these baseline metrics. SWE-Hard achieves an mAP of 0.776, an MRR of 0.811, and a Recall@20 of 0.931. This indicates the potential of a very high-performing QbE audio search system based on word embeddings.

Table 4.3: Retrieval evaluation results for the Swedish pretrained model. Comparison between original and transformed embeddings for the Swedish test dataset.

Model	**mAP**	**MRR**	**Recall@20**
VoxRex (Original)	0.440	0.469	0.602
SWE-Hard	**0.777**	**0.812**	**0.932**
percent increase	76.5	73.1	54.8

In Table 4.4, information retrieval metrics are displayed. The starting point of the original XLS-R embeddings was quite poor. This meant the relative performance increase of X-Hard transformed embeddings was larger compared to SWE-Hard. In absolute terms, mAP and MRR are about half, and Recall@20 is about $2/3$ of SWE-Hard's QbE performance metrics. This illustrates that X-Hard is having a hard time retrieving the correct word embeddings. Although it seemed like word clusters were well separated in Figure 4.4, there were only a few words sampled and projected. By looking at more metrics as training loss 4.2, and original UMAP clustering 4.4 (left), it seems like the cross-lingual embeddings are being similarly encoded.

Table 4.4: Retrieval evaluation results for the Multilingual model XLS-R. Comparison between original and transformed embeddings.

Model	**mAP**	**MRR**	**Recall@20**
XLS-R (Original)	0.149	0.159	0.282
X-Hard	**0.385**	**0.424**	**0.676**
percent increase	159.1	165.9	139.6

Search Results of Selected Queries

Some qualitative examples for a set of queries and the database responses are next up in the evaluation. In Table 4.5, five Swedish queries belonging to the test set are used for vector search of the FAISS database. The top 5 results are returned with their associated cosine similarity. On the left, VoxRex embeddings are used for querying the database of word embeddings. On

the right, transformed embeddings using SWE-Hard are returned with cosine similarities using a transformed query. The transformed embeddings perform better at retrieving more samples of the same word from the database. It is also evident that phonetically similar words are, to some extent, returned among the top 5. Meaning that the embedding space is structured to place phonetically similar utterances close together in the embedding space. This indicated that speaker characteristics and background noise have reduced importance in the embedding similarity. Transformed embeddings seem to encode phonetic content to a larger degree compared to native embeddings.

Table 4.5: Original and transformed embedding retrieval results per query word for the SWE-Hard embedding projection.

Query	**Retrieved (Native)**	**Retrieved (Transformed)**
talare	talare (0.975)	talare (0.978)
	talare (0.964)	talare (0.941)
	avsaknad (0.957)	talare (0.917)
	förare (0.956)	talare (0.883)
	göras (0.953)	skadar (0.845)
slutändan	slutändan (0.986)	slutändan (0.923)
	regelbunden (0.963)	slutändan (0.911)
	nedsättande (0.952)	slutändan (0.883)
	avslöjandet (0.949)	söndag (0.828)
	natten (0.948)	ända (0.799)
aggressiva	aggressiva (0.960)	aggressiva (0.942)
	byggstenarna (0.934)	aggressiva (0.939)
	insatserna (0.933)	aggressiva (0.857)
	vanliga (0.932)	civiliserad (0.677)
	vandringsleder (0.930)	legalisering (0.664)
jämföra	ordförande (0.955)	jämföra (0.929)
	ordförande (0.953)	jämföra (0.849)
	besökare (0.950)	jämföra (0.789)
	underförstådda (0.950)	snöbroar (0.773)
	oönskade (0.948)	föreslår (0.749)
buffalo	varning (0.939)	klagomål (0.787)
	bära (0.937)	varning (0.774)
	fallit (0.937)	klagomål (0.773)
	upplever (0.937)	klagomål (0.758)
	sådana (0.936)	rovfågel (0.742)

The multilingual model is evaluated with the multilingual dataset. Five

example queries and the top 5 closest neighbors in the embedding space are displayed in Figure 4.6. These qualitative examples show that the multilingual model is struggling with retrieving the correct samples from the multilingual database, yielding a very low QbE performance. When taking a closer look at each query, it could be that the transformed embedding space is structured by language, and not by phonetic similarity. The Finnish word alkoi is in both original embeddings and transformed, most similar to other, non-phonetically similar Finnish words. The same observation is made for the query efterfølgende, a Danish word which is transformed to be more similar to other Danish words that lack phonetic similarity.

Table 4.6: Original and transformed embedding retrieval results per query word for the X-Hard embedding projection.

Query	**Retrieved (Native)**	**Retrieved (Transformed)**
alkoi	valonsädettä (0.994)	alkoi (0.880)
	voitti (0.993)	julkaisi (0.872)
	hautajaismatkat (0.992)	voitti (0.862)
	liberale (0.992)	ruokkia (0.857)
	högre (0.992)	alkoi (0.852)
redigera	etiopien (0.996)	européer (0.867)
	assyriska (0.995)	lokale (0.828)
	covid19viruset (0.995)	indikerar (0.800)
	lahjoituksia (0.995)	redigera (0.800)
	beauty (0.995)	byggarbetet (0.799)
efterfølgende	evakuerats (0.993)	tommelfingerregel (0.901)
	ansvar (0.993)	opplevelsen (0.899)
	jurister (0.993)	prøvekaraktererne (0.883)
	behandling (0.992)	medførte (0.871)
	toimesta (0.992)	lagde (0.859)
nærmest	baseres (0.997)	nærmest (0.890)
	andreplass (0.997)	tok (0.856)
	årene (0.996)	danius (0.814)
	anslöt (0.996)	leirseng (0.808)
	mer (0.996)	estland (0.772)
parallelle	gjerdet (0.996)	celler (0.849)
	eventuelle (0.995)	verdeøyene (0.842)
	liberaalien (0.995)	føderale (0.837)
	genvalget (0.995)	generellt (0.817)
	umiddelbart (0.995)	sverige (0.793)

4.4 Discussion

Here, the results are discussed with some reflections about ways of improvement and the probable causes of the experimental outcomes.

4.4.1 Language-Specific Pre-Training Drives Performance

The monolingually pre-trained VoxRex model outperformed the multilingually pre-trained XLS-R model. Factors like initial embedding space structure of the models, the experimental setup in this work, and training dataset distribution could all be contributing to this result. The Swedish model used around 10,000 Swedish speech samples for training, testing, and validation. In contrast, the XLS-R-based model used around 8,000 total samples, stratified by the four languages. This, as well as the triplet dataset structure, probably contributed greatly to the performance difference.

The greatest, and only difference of these pre-trained models is what they are trained on. VoxRex has its embedding space specialized to model Swedish speech; it uses its full capacity, all 300 million parameters, for Swedish phonetic representations. XLS-R, which is also a 300 million parameter model, is pre-trained in 128 languages, and has to accommodate a larger portion of the embedding space to many varying speech representations.

VoxRex-based Embeddings are Generalizing

VoxRex, which is pre-trained in varied Swedish speech, shows great potential with a minimal downstream training setup. It achieved high clustering scores and competitive information retrieval metrics on the test set, which contains words unseen during training. This shows that a small network, using around 6,000 unique samples, of which 1,800 were unique words, trained for only a few minutes to solve a contrastive task, has high potential for QbE audio search.

XLS-R-based Embeddings are Struggling to Transform

For Swedish, Danish, Norwegian, and Finnish, the model began clustering acoustic word embeddings, but not sufficiently for robust QbE audio search. The results suggest that the X-Hard model has started encoding language-specific features rather than phonetic content. This is expected given the construction of the multilingual triplet dataset: anchor and positive samples

always come from the same language, while negatives can come from any language. This setup reinforces clustering by language rather than by phonetic similarity. Additionally, consistent audio quality within each language subset of FLEURS may lead to language-specific background noise being encoded in the embeddings.

4.4.2 Structuring the Embedding Space for Phonetic Relevance

The intended function of the embedding transformation network was to act as a filter that removed unwanted features from a single word utterance. A contrastive objective that lets a network cluster samples based on shared phonetic content, removing factors like noise and speaker identity, has shown promise for the Swedish model. However, in the multilingual experimental setup, always training with anchor and positive samples from the same language, limited the effectiveness of contrastive learning. If we aim to shape the embedding space to align phonetically, gradient-like, there is a need for nuanced and varied training triplets, created for phonetic similarity.

The challenge is to construct an objective that promotes the ideal embedding structure; same words cluster tightly, phonetically similar words are placed reasonably close, and completely dissimilar words are placed far away.

Triplet margin loss is binary; it takes two classes and pushes them apart despite classes having varying similarity. In our case, where our objective is to construct a smooth space with graded transitions, triplet loss could be too rigid. Other approaches, such as the use of a softer margin or even a quadruplet loss, should be looked into. It could also be beneficial to construct a loss function that considers phonetic distance to align the training objective with the downstream task directly.

4.4.3 The Hard Triplet Problem

The purpose of hard triplets is to train the model to distinguish between highly similar samples. In this work, we aim to create a gradient-like phonetic embedding space, where similar utterances are placed closely, but not overlapping. Having hard triplets based on cosine similarity could be counterproductive for our goal, especially if the loss function margin parameter is fixed during training. Having the same margin across training despite varying phonetic similarity of positives and negatives could be a

limiting factor. This work employs margin, hard triplet ratio, and hard triplet cosine similarity annealing, meaning that the margin gets progressively higher and more hard triplets are used over epochs. The motivation behind it is that we first want a global structure of embeddings by having low requirements for separation, in later epochs, we want fine-grained clustering, so we increase the margin. By looking at Figure 4.2, which shows training loss, validation score, and annealing schedule for the multilingual model. The validation score drops, and the model seems to worsen with this annealing setup. This could be a clue to understanding triplet loss and how it transforms embeddings. In future work, an opposite annealing approach could be employed to work towards optimal embedding space transformation.

Cosine similarity is used to measure embedding similarity when sampling hard triplets. It compares the original embeddings, which we know already have some phonetic similarity, but they also encode irrelevant information. By using this similarity to samples hard negatives, some might be beneficial for our objective by being non-phonetically similar, but highly similar in irrelevant characteristics. In some cases, a negative with higher phonetic similarity is chosen as a negative and pushed away in the backpropagation step, which could confuse the model or make it unstable.

4.4.4 Towards a Fully Unsupervised Pipeline

In application, a fully unsupervised QbE audio search pipeline needs two stages. This work proposes the second stage, which is embedding transformation and vector searching. The initial step concerns unsupervised word segmentation of the long-format audio that is searched through. Since embeddings are trained on a word level, the input audio and query used for searching need to be word segments. Some methods should be looked into for this. An ensemble of sliding windows of varying resolutions could be used to segment speech signals. This is considered a low-level approach, but could be a viable starting point. Sophisticated approaches with neural networks, such as Segmental Contrastive Predictive Coding [27], exist. It uses a convolutional neural network to learn frame-level representations from a raw waveform, which are used to optimize a segment encoder with noise-contrastive estimation. Producing both phoneme and word segmentation that outperforms existing methods. This, as stated in the introduction, constitutes an entire thesis itself and is highly encouraged for future work.

Chapter 5

Conclusions

5.1 Summary of Contributions

This thesis presents a robust Query-by-Example pipeline built upon wav2vec 2.0 embeddings transformed through triplet-loss-based contrastive learning. The contributed network and training setup showed that the original wav2vec 2.0 embeddings could be transformed to improve clustering of the same word utterances across different speakers and recording conditions. A metric-based evaluation with relevant information retrieval benchmarking showed that word-segmented queries are powerful in creating a phonetically structured embedding space that disregards speaker-specific features and background noise. Using fixed-size word embeddings also facilitates near-instant vector search that can scale well.

The monolingually pre-trained Swedish VoxRex wav2vec 2.0 model significantly outperformed the cross-lingual XLS-R model in the QbE audio search task. This leads us to conclude that pre-training and language specificity play a major role in the ability of large transformers to be used for representation learning, given this work's experimental setup.

5.2 Future Work

There were many paths of improvement explored in the discussion. The two primary subjects for large improvement and further research are dealing with the construction of the contrastive objective function, so that it is goal-oriented to reach the desired augmentation of the embedding space, and the hard triplet dataset sampling considerations.

When aiming to create an embedding space that has smooth, graded

transitions of similar classes, a soft or gradient-based objective function should be employed. In this work, the triplet margin loss was used, and results have indicated that it might be too rigid and binary for the desired outcome.

The triplet training dataset was, of course, a requirement to train a network with this objective function. When sampling triplets that are supposed to contribute to the objective, there was no consideration of the confusion that phonetically similar ”hard” negative samples could cause.

We suggest that future work should pay specific attention to three subjects: (1) using a different objective function that can cluster classes more carefully and with phonetic similarity built into the loss function for a gradient-like embedding space. (2) When using triplet loss, focus experiments on constructing training data of the highest quality. Using synthetic data is a powerful approach that should give the researcher control and knowledge of the training data distribution. (3) To complete the pipeline proposed in this work, an unsupervised word-segmentation method should be created and incorporated into this pipeline. It is the missing piece before going completely transcript-free in the audio search domain, which is highly relevant to the industry because of the computation time and cost of transcription-based search.

5.3 Reflections

Sustainability

Speech technology has several economic, social, and environmental sustainability implications. The objective of this work is to further research transcription-free audio search. Moving away from transcription-based ASR would primarily be economically beneficial, since labeling audio with accurate transcriptions is costly and time-consuming.

In a low-resource setting, having a single multilingual model that can detect phonetically similar segments is beneficial. An acoustic approach that is language agnostic would give broader access to speech recognition, driving inclusivity to underrepresented languages. This is why a cross-lingual model is evaluated in this work, to determine whether pre-trained models that are created for low-resource settings can be used in a multilingual representation learning application. In this experimental context, there is high promise shown, but a lot of work is still needed to achieve good performance.

The energy consumption of training transformer-based embedding models, such as wav2vec 2.0, is high. In this work, we leveraged two pre-

trained, publicly available wav2vec 2.0 models. The ASR community should focus on using available models to avoid wasting energy pre-training similar models. Also, transcribing and prompting large language models to process transcriptions is costly. Using frozen embeddings from pre-trained models and training a small network on top is more energy-efficient than end-to-end ASR pre-training.

Ethics

When working with personal data such as speech audio, it is important to adhere to both legal and moral obligations. Speech data can considered personal data, as it may contain information about a speaker's identity or accent. In this work we have used the Google FLEURS dataset which is distributed under the Creative Commons Attribution (CC-BY) license, meaning it permits use, distribution, and modification, provided that the original authors are credited. It also means that speakers have given their consent for their recordings to be used for research and development.

This thesis focuses on unifying different word pronunciation to create embeddings that have eliminated any personal variations. Therefore, the ethical risks of this work is minimal. Also, it does not collect new personal speech data nor process any data that is considered sensitive personal information.

Societal Aspect

This work contributes to general knowledge of ASR, audio search, low-resource language, and representation learning. Hopefully, authors of future work and the ASR community can benefit from this thesis and further develop the path towards transcription-free audio search. The results show the high potential of transforming frozen embeddings from a large public model to skip the costly step of pre-training.

References

[1] L. Mary and D. G, "Audio search techniques," in *Searching Speech Databases: Features, Techniques and Evaluation Measures*, L. Mary and D. G, Eds. Springer International Publishing, pp. 1–12. ISBN 978-3-319-97761-4. [Online]. Available: https://doi.org/10.1007/978-3-319-97761-4_1 [Pages 1, 11, and 14.]

[2] S. Liu, A. Mallol-Ragolta, E. Parada-Cabaleiro, K. Qian, X. Jing, A. Kathan, B. Hu, and B. W. Schuller, "Audio self-supervised learning: A survey," vol. 3, no. 12, p. 100616. doi: 10.1016/j.patter.2022.100616. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2666389922002410 [Page 1.]

[3] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations." [Online]. Available: http://arxiv.org/abs/2006.11477 [Pages 2 and 20.]

[4] M. Malmsten, C. Haffenden, and L. Börjeson, "Hearing voices at the national library – a speech corpus and acoustic model for the swedish language." [Online]. Available: http://arxiv.org/abs/2205.03026 [Pages 2, 21, and 27.]

[5] A. Conneau, M. Ma, S. Khanuja, Y. Zhang, V. Axelrod, S. Dalmia, J. Riesa, C. Rivera, and A. Bapna, "FLEURS: Few-shot learning evaluation of universal representations of speech." [Online]. Available: http://arxiv.org/abs/2205.12446 [Pages 2 and 21.]

[6] R. Ardila, M. Branson, K. Davis, M. Henretty, M. Kohler, J. Meyer, R. Morais, L. Saunders, F. M. Tyers, and G. Weber, "Common voice: A massively-multilingual speech corpus," in *Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020)*, 2020, pp. 4211–4215. [Page 2.]

[7] S. Barrington, R. Barua, G. Koorma, and H. Farid, "Single and multi-speaker cloned voice detection: From perceptual to learned features." [Online]. Available: http://arxiv.org/abs/2307.07683 [Page 2.]

[8] G. Deekshitha and L. Mary, "Multilingual spoken term detection: a review," vol. 23, no. 3, pp. 653–667. doi: 10.1007/s10772-020-09732-9. [Online]. Available: https://doi.org/10.1007/s10772-020-09732-9 [Page 2.]

[9] M. H. Moattar and M. M. Homayounpour, "A simple but efficient real-time voice activity detection algorithm," in *2009 17th European Signal Processing Conference*, pp. 2549–2553. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7077834 [Page 2.]

[10] A. Plaquet and H. Bredin, "Powerset multi-class cross entropy loss for neural speaker diarization," in *INTERSPEECH 2023*. ISCA. doi: 10.21437/Interspeech.2023-205 pp. 3222–3226. [Online]. Available: https://www.isca-archive.org/interspeech_2023/plaquet23_interspeech.html [Page 2.]

[11] J. Li, A. Sun, J. Han, and C. Li, "A survey on deep learning for named entity recognition," vol. 34, no. 1, pp. 50–70. doi: 10.1109/TKDE.2020.2981314 Conference Name: IEEE Transactions on Knowledge and Data Engineering. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9039685 [Page 2.]

[12] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives." [Online]. Available: http://arxiv.org/abs/1206.5538 [Page 8.]

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need." [Online]. Available: http://arxiv.org/abs/1706.03762 [Pages ix, 9, and 10.]

[14] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," vol. 20, pp. 53–65. doi: 10.1016/0377-0427(87)90125-7. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0377042787901257 [Page 13.]

[15] W. M. Rand, "Objective criteria for the evaluation of clustering methods," vol. 66, no. 336, pp. 846–850. doi: 10.1080/01621459.1971.10482356 Publisher: ASA Website _eprint:

https://www.tandfonline.com/doi/pdf/10.1080/01621459.1971.10482356. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/01621459.1971.10482356 [Page 13.]

[16] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: is a correction for chance necessary?" in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09. Association for Computing Machinery. doi: 10.1145/1553374.1553511. ISBN 978-1-60558-516-1 pp. 1073–1080. [Online]. Available: https://doi.org/10.1145/1553374.1553511 [Page 13.]

[17] L. McInnes, J. Healy, and J. Melville, "UMAP: Uniform manifold approximation and projection for dimension reduction." [Online]. Available: http://arxiv.org/abs/1802.03426 [Page 13.]

[18] A. E. L. Dahlin, *Attribute Embedding for Variational Auto-Encoders : Regularization derived from triplet loss*. KTH Royal Institute of Technology. [Online]. Available: https://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-321530 [Page 15.]

[19] M. Rivière, A. Joulin, P.-E. Mazaré, and E. Dupoux, "Unsupervised pretraining transfers well across languages." [Online]. Available: http://arxiv.org/abs/2002.02848 [Page 16.]

[20] S. Settle, K. Levin, H. Kamper, and K. Livescu, "Query-by-example search with discriminative neural acoustic word embeddings." [Online]. Available: http://arxiv.org/abs/1706.03818 [Page 16.]

[21] Y. Hu, S. Settle, and K. Livescu, "Acoustic span embeddings for multilingual query-by-example search." [Online]. Available: http://arxiv.org/abs/2011.11807 [Page 16.]

[22] S.-w. Yang, P.-H. Chi, Y.-S. Chuang, C.-I. J. Lai, K. Lakhotia, Y. Y. Lin, A. T. Liu, J. Shi, X. Chang, G.-T. Lin, T.-H. Huang, W.-C. Tseng, K.-t. Lee, D.-R. Liu, Z. Huang, S. Dong, S.-W. Li, S. Watanabe, A. Mohamed, and H.-y. Lee, "SUPERB: Speech processing universal PERformance benchmark." [Online]. Available: http://arxiv.org/abs/2105.01051 [Pages 16 and 20.]

[23] A. Babu, C. Wang, A. Tjandra, K. Lakhotia, Q. Xu, N. Goyal, K. Singh, P. v. Platen, Y. Saraf, J. Pino, A. Baevski, A. Conneau, and M. Auli,

"XLS-r: Self-supervised cross-lingual speech representation learning at scale." [Online]. Available: http://arxiv.org/abs/2111.09296 [Pages 21 and 27.]

[24] M. Bain, J. Huh, T. Han, and A. Zisserman, "WhisperX: Time-accurate speech transcription of long-form audio." [Online]. Available: http://arxiv.org/abs/2303.00747 [Page 22.]

[25] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision." [Online]. Available: http://arxiv.org/abs/2212.04356 [Page 22.]

[26] M. Douze, A. Guzhva, C. Deng, J. Johnson, G. Szilvasy, P.-E. Mazaré, M. Lomeli, L. Hosseini, and H. Jégou, "The faiss library." [Online]. Available: http://arxiv.org/abs/2401.08281 [Page 27.]

[27] S. Bhati, J. Villalba, P. Żelasko, L. Moro-Velazquez, and N. Dehak, "Segmental contrastive predictive coding for unsupervised word segmentation." [Online]. Available: http://arxiv.org/abs/2106.02170 [Page 39.]

TRITA-EECS-EX-2025:546
Stockholm, Sweden 2025

www.kth.se