# Aligning Self-supervised Speech Representations for Different Speakers via Linear mapping

**B196193**

## Abstract

Contrastive predictive coding (CPC) models can extract useful representations from speech audio. These representations can achieve high accuracy in speaker and phone prediction, and can improve the performance of speech processing systems. We are interested in how speaker and phone information are organised in CPC representations, as well as methods for removing speaker information, since many spoken language modeling tasks benefit from speaker-invariance. We computed a vector for each phone in each speaker space as the mean of all CPC vectors of the same label and produced by the same speaker. We found that these phone vectors can be closely aligned across speakers and the alignment mapping can be applied to all CPC representations of that speaker to effectively eradicate speaker information while maintaining most phone information in them.

wanted a more concise summary of the method that does not mention phone vectors but could not think of one...

## 1 Introduction

Self-supervised learning models are developed to extract useful representations from unlabelled data. Generally speaking, these models are trained to predict samples of the unlabelled data given other samples from neighbouring time-steps, and the representations extracted from the models can be used to bootstrap training of downstream systems. In the speech domain, these representations have proved to improve performances in automatic speech recognition (van den Oord et al., 2019; Schneider et al., 2019) and help reduce reliance on annotated training data, which is expensive to create and particularly scarce for low-resource languages.

create doesn't seem to be the right word here

Apart from practical applications, self-supervised speech models can potentially provide insights on the mechanism for early language acquisition, namely how infants learn language by being exposed to its spoken form. Linguists have found that infants' perception of speech sound contrasts changes significantly during the first year of life. Specifically, their discrimination of native sound contrasts improves, and discrimination of non-native ones declines (Kuhl et al., 2006). Researchers have proposed that this perceptual change could be driven by unsupervised representation learning and called for unsupervised representation learning algorithms developed in the machine learning community to be examined for their cognitive relevance and plausibility (Feldman et al., 2021).

Previous research has developed a range of self-supervised speech models with different architectural design and choice of learning signals (Liu et al., 2020; Chung et al., 2019, *inter alia*) . In this work, we focus on the Contrastive Predictive Coding (CPC) model proposed by van den Oord et al. (2019). Given a segment in an audio sequence, CPC is trained to distinguish a future sample in the same sequence from a set of negative samples drawn from a different segment in the sequence or other sequences. Representations extracted from modified versions of CPC have been shown to improve the performance in monolingual automatic speech recognition on several benchmarks (Schneider et al., 2019), and also transfers well in cross-lingual settings (Riviere et al., 2020). Moreover, linear classifiers trained on CPC representations achieved high accuracy in phone classification (van den Oord et al., 2019), even approaching that of supervised representations in speaker classification. Finally, Blandón and Räsänen (2020) found that the CPC model rapidly converged to phoneme-discriminative representations during early training epochs, achieving good performance in the ABX phone discrimination test (Schatz et al., 2013) after only one pass of the training data. The rapid convergence of CPC resembles child language acquisition to a greater extent than other models that often require multiple iterations over the training set to converge.

Despite the usefulness of CPC representations, it remains to be understood how phonetic and speaker information are organised in them and whether the two can be disentangled. Recently, van Niekerk et al. (2021) found that per-utterance means of CPC features are informative of speaker identities. They then proposed to perform speaker normalisation for CPC representations by applying utterance-level standardisation as a post-processing step. This boosted phone classification accuracy from 75.7% to 77.0% and reduced speaker classification accuracy from 93.4% down to 14.8%. They also observed minor improvements in within- and across-speaker ABX scores, as well as in performance on spoken language modeling tasks.

same problem as in the abstract, struggled to distill central points concisely

In this work, we found that *mean phone vectors* can be closely aligned across speakers. Each of these vectors are aggregated from CPC representations sharing the same phone label and extracted from each speaker. Taken together, the 39 mean vectors corresponding to 39 phone labels characterise the CPC phone space of a speaker. We applied the linear mapping, as computed from aligning the mean phone vectors of two speakers, to align all CPC representations of the two speakers, hence removing speaker-specific information in those CPC representations. When extended beyond two speakers to a speaker normalisation setup, this method reduced the probing accuracy of speaker identity from 99.5% to 4.6% while causing a minor drop in phone probing accuracy (76.0% to 72.3%). At the end of this report, we discuss the possibilities of replacing the mean phone vectors, which requires time-aligned transcriptions to compute, with eigenvectors of the speaker space or representations of the same word or utterance produced by different speakers, hence making our method unsupervised.

## 2 Analysing CPC Representations

### 2.1 Contrastive Predictive Coding

The CPC model consists of an encoder $g_{enc}$ and an auto-regressive context summariser $g_{ar}$. The encoder maps each input frame to a latent embedding $z_t = g_{enc}(x_t)$. The latent embeddings up until time $t$ are summarised to produce a context embedding $c_t = g_{ar}(z_1, z_2, ..., z_t)$. Given the context embedding $c_t$, the model is trained to predict the next $M$ latent embeddings $\{z_t + m\}_{1 \leq m \leq M}$ by minimising the following contrastive loss:

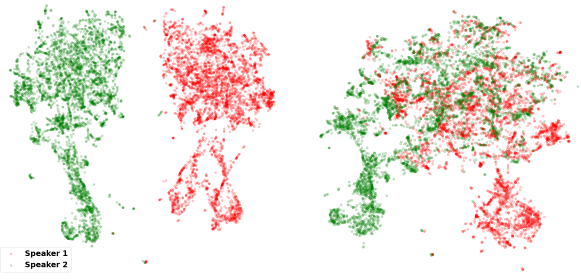
$$\mathcal{L} := -\frac{1}{M}\sum_{m=1}^{M}\log\left[\frac{\exp(z_{t+m}^{\mathbf{T}} W_m c_t)}{\sum_{\tilde{z}\in\mathcal{N}_t}\exp(\tilde{z}^{\mathbf{T}} W_m c_t)}\right]$$

$\mathcal{N}_t$ is a set of negative embedding samples, and $W_m$ is a linear mapping for projecting context embeddings into the embedding space of $z_t$.

In this project, we followed van Niekerk et al. (2021) and focused on representations extracted from the CPC-big model, which was used as the baseline in the 2021 ZeroSpeech challenge (Nguyen et al., 2020). The encoder is made up of 5 1D-convolutional layers with kernel sizes of 10,8,4,4,4 and stride sizes of 5,4,2,2,2, while the context summariser consists of 4 LSTM layers, each with 512 hidden units. The prediction horizon $M$ is set to 12 time-steps, i.e. 120 ms. The raw audio is sampled at 16 kHz and the hop between each latent embedding $z_t$ is 10 ms. This implementation of CPC followed the original CPC (van den Oord et al., 2019) in sampling negative samples within the same speaker, although $\{W_m\}_(1 \leq m \leq M)$ is operationalised with a Transformer network instead of a linear layer. The model is trained on LibriLight `unlab-6k` set (Kahn et al., 2020), which includes 5770 hours of read speech. Throughout our analysis, we use the representations extracted from the second LSTM layer in the context summariser, since Nguyen et al. (2020) they gave the best results in ABX tests in Nguyen et al. (2020).

### 2.2 Visualising the Effect of Standardisation

van Niekerk et al. (2021) applied dimensionality reduction technique and found that per-utterance means computed from CPC representations of the same speaker form clusters in the representation space, and those for different speakers are clearly separated. They also visualised CPC representations extracted from all the speech data of two speakers, and showed that the overlap between them increased significantly after performing utterance-level standardisation.



(a) Original representations (b) After standardisation

Figure 1: UMAP visualisation of all CPC representations for two speakers in the `dev-clean` subset of LibriSpeech (speaker 1 ID: 1993, speaker 2 ID: 652)

We observe the same phenomenon in our replicated plot (Figure 1), where each dot in the figure denotes one CPC vector ($c_t$). We took interest in the two bottom clusters of the two speakers which resembled each other in shape but remained well-separated after standardisation. This motivated us to study whether additional mapping operation on the representations, *e.g.* linear transformation, can further improve the degree of overlap between speaker spaces.

Should I define speaker spaces?
Is it self-explanatory given the context or is it vague or confusing?

One thing to note is that our objective is not to have the CPC representations in the plot to overlap completely. This is because the speech produced by the two speakers, from which we extracted the representations, do not share the same content, although we might expect the occurrence frequencies of each phoneme to be similar between the two speakers when the speech samples are both long enough. This makes it hard to interpret how much speaker information is removed simply from the extent that the extracted representations are overlapped, hence calls for more fine-grained analysis of the representation space.

### 2.3 Distribution of Mean Phone Vectors

To gain more insights into the two bottom clusters as well as the rest of the representation space, we performed a more detailed analysis by grouping the CPC vectors by the phonetic category that they correspond to. With forced alignment techniques, we aligned each CPC vector with a phone label in the transcription of the speech and then computed a *mean phone vector* for each phone label from all the CPC vectors with that specific label, and analyse how these mean phone vectors are distributed within and across speakers.

For visualisations, we followed van Niekerk et al. (2021) in using UMAP, a manifold learning dimensionality reduction technique (McInnes et al., 2018). Compared to T-SNE, UMAP not only preserves local structure of the high-dimensional space just as well, but also maintains a better global structure, *i.e.* inter-cluster distances (Oskolkov, 2021). This merit of UMAP makes it well-suited for our purpose of observing the relative position of vector clusters corresponding to different phone labels.

In Figure 2, we present visualisations of mean phone vectors computed from CPC representations before utterance-level standardisation. In these plots, the marker for each mean phone vector is the ARPABET symbol of the corresponding phone label [1]. Looking at Figure **??**, we can see that the bottom clusters in Figure 1 that could not be aligned with standardisation were actually silences ('SIL'). We also observe that phones with similar properties tend to appear closer in the representation space, daas we would expect. For instance, we can see the following clusters of consonants (with IPA symbols in the brackets):

- Nasals: 'N' (n), 'M' (m), 'NG' (ŋ)
- Affricates: 'CH' (tʃ), 'SH' (ʃ), 'JH' (dʒ)
- Fricatives: 'S' (s), 'Z' (z)

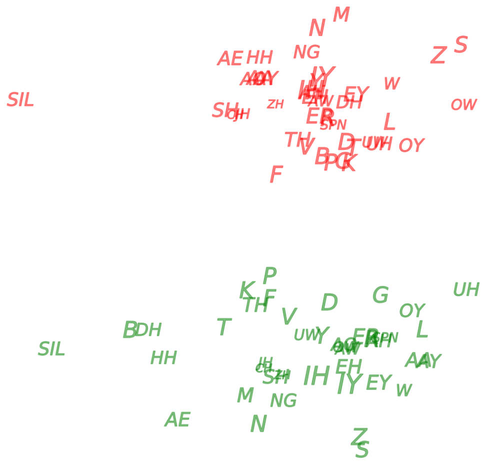


Figure 2: Mean phone vectors computed from CPC features before processing

To see the effect of standardisation, we compare the mean phone vectors before and after standardisation in plots (a) and (b) of Figure 3 (for consonants) and Figure 4 (for vowels). Although standardisation effectively aligned certain consonants (ɡ, k, f, v, ŋ, h) and vowels (ʌ, ɔ, ɪ, oʊ, ʌɪ), it failed for the remaining 28 phones, most notably for affricates and fricatives.

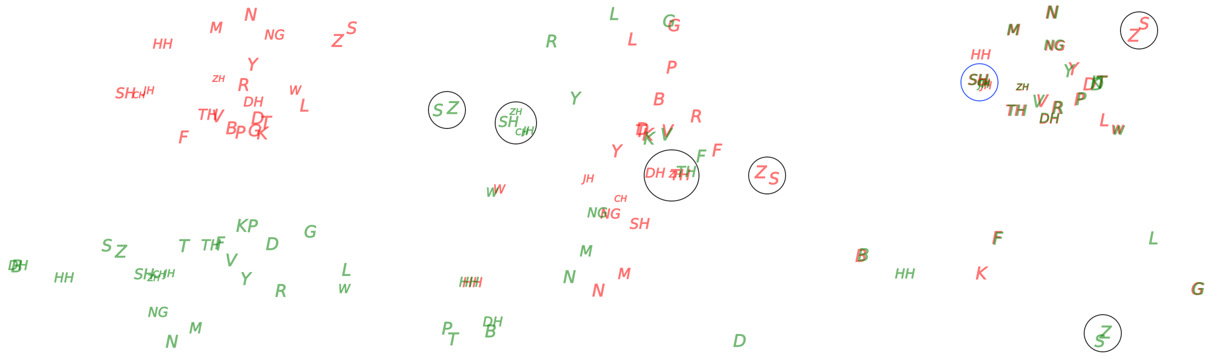
To summarise our findings from this visual exploration: we found that the two bottom clusters that did not overlap after standardisation were in fact silences. The rest of the CPC representations of the two speakers seem to overlap to a large extent after standardisation. However, after grouping the vectors by their corresponding phone labels and representing each phone label with a mean vector, we found that 28 out of 39 phones were far from aligned. In particular, standardisation had little effect in aligning affricates and fricatives.

## 3 Removing Speaker Information via Linear mapping

Our hypothesis is that how different phones are organised relative to each other in the representation space is similar across speakers. This would

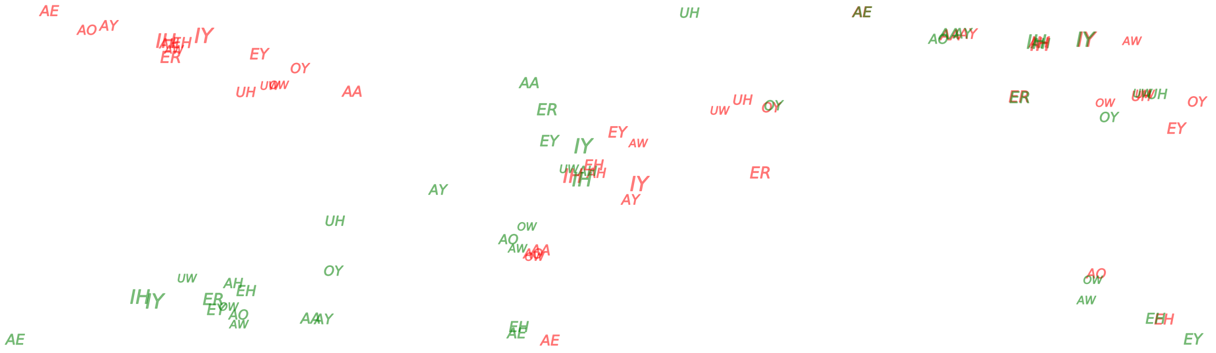
Does this sentence make sense?

[1] The correspondence between ARPABET and IPA symbols can be found at https://en.wikipedia.org/wiki/ARPABET



(a) Original representations (b) After standardisation (c) After linear mapping

Figure 3: Mean phone vectors for consonants before and after processing



(a) Original representations (b) After standardisation (c) After linear mapping

Figure 4: Mean phone vectors of the same two speakers before and after the post-processing steps.

mean that we could use phone information to align representation spaces of different speakers, hence reducing speaker-specific information in the representations. In this work, we focus on applying linear mapping to align the representation spaces for a pair of speaker. This could be easily extended to speaker normalisation for multiple speakers by aligning all speakers to a single anchor speakers.

**Task Definition** From speech audios of an anchor speaker and a test speaker, we extract CPC representations and then concatenate the representations of the anchor speaker to form matrix $X_a$, and those of the test speaker to form $X_t$. Each row in $X_a$ and $X_t$ corresponds to a CPC vector and has a dimension of 512, but in most cases the two matrices would not have the same number of rows because the speech audio of the two speakers are unlikely to have exactly the same duration. The objective is to find a mapping that maps $X_t$ to $X_t'$ such that vectors corresponding to the same labels within $X_a$ and $X_t'$ are maximally similar.

**Computing the Alignment mapping** Since our objective is not to align $X_a$ and $X_t$ per se, we need to extract some common features or structure shared by the two speakers from the data, and align the matrices representing those features or structures instead. Say that we extract from $X_a$ and $X_t$ two sets of common features, represented as matrices $Y_a$ and $Y_t$. We can then compute a mapping that maximally aligns them, $M = \arg\min_M ||Y_t \cdot M - Y_a||$. When $M$ is constrained to be an orthogonal matrix, it has a closed-form solution: $M = VU^T$, where $U$ and $V$ are resulted from applying singular value decomposition to $Y_a Y_t^T$ $(Y_a^T Y_t = U\Sigma V^T)$.

'common features or structures' don't exactly express what I try to get at

**Phone Vectors as the Common Feature for Alignment** Recall that we hypothesised that representations of phones are organised similarly across speakers. If this were true, we can extract representations of a common set of phones from $X_a$ and $X_t$, and use them to compute the alignment mapping. For example, having extracted a vector that represents phone s produced by the anchor speaker from $X_a$ and a vector for s speaker from $X_t$, we would try to maximally align these two vectors, as well as other pairs of vectors representing the same phone for the two speakers.

Before using phone vectors as the common feature for computing the alignment mapping, we would need to evaluate our hypothesis, *i.e.* that they *can* indeed be aligned. For this purpose, we concatenated the aforementioned mean phone vectors for each speaker into a matrix and use this to characterise the phone space of each speaker. The rows in each matrix are arranged corresponding to a fixed order of phone labels. We measure how closely the phone spaces of two speakers are or can be aligned by computing the average cosine similarity between rows with the same index of any two such matrices. The average similarity is 0.59 for the original CPC representations before any post-processing and goes up to 0.92 after standardisation. If we align the two matrices with the linear mapping method mentioned above, the similarity becomes 0.99. On the other hand, if we shuffle the rows of the matrices before computing the alignment between them, the similarity after alignment is only 0.71. In other words, the phone space matrices can be almost perfectly aligned for two speakers if and only if each row is to be aligned with a row that corresponds to the same phone. We can also see this visually in Figures 3 (c) and 4 (c). While this method still fails to align the fricatives sounds s and z, most other mean phones vectors overlapped to a great extent. These results and observations provide some support for our hypothesis and drive us to explore whether applying the alignment mapping, as computed from mean phone vectors, to all CPC representations can effectively remove speaker information in them. It is worth noting that the mean phone vectors can only be obtained when the transcriptions are available and we will discuss possibilities for obtaining representations of the phone space without supervision in section 6.

## 4 Evaluation

**ABX test** The machine ABX task (Schatz et al., 2013) is used to test the discrimination of phonetic contrasts in speech representations. Each test sample in an ABX task is a triplet of triphone tokens (A, B, X), *e.g.* (A = beg, B = bag, X = bag$^{'}$). All three triphones represented share the same left- and right-context (*i.e.* b*g), with B and X being different tokens of the same triphone, and A having a different centre phoneme. In other words, the triphone of A forms a minimum pair with that of B and X. Given such triplets, all we need to run the test is a similarity metric between representations of the three triphone tokens. Denoting sim(X, Y) as the similarity value between the representations of X and Y,

$$\text{score} = \begin{cases} 1 & \text{if sim(B, X)} > \text{sim(A, X)} \\ 0 & \text{if sim(B, X)} < \text{sim(A, X)} \\ 0.5 & \text{if sim(B, X)} = \text{sim(A, X)} \end{cases}$$

The final ABX score is reported after aggregating over all minimum pairs and all token combinations of each minimum pair. For *within speaker* ABX task, all three tokens in the triplet were produced by the same speaker, *e.g.* A = beg$_{S1}$, B = bag$_{S1}$, X = bag$^{'}_{S1}$. For *across speaker* ABX, A and B are are different tokens produced by the same speaker, with X being produced by another speaker, *e.g.* A = beg$_{S1}$, B = bag$_{S1}$, X = bag$_{S2}$.

Since a CPC model extracts frame-level representations from speech segments, each one of A, B, X would correspond to not a single, but a sequence of vectors. Also, A, B, X might differ in their exact durations and hence represented with vector sequences of different lengths. To compare the similarity between them, dynamic time warping is performed between the two representation sequences before computing the average over frame-wise cosine similarity to give sim(X, Y). We used the scripts provided by the ZeroSpeech 2021 challenge to run the ABX task on te dev-clean subset of the LibriSpeech dataset (Panayotov et al., 2015).

While we expect better phonetic discriminability to be correlated with speaker-invariance, the ABX test is not directly targeted at testing speaker information in representations. Hence, we also implemented probing experiments to evaluate the linear separability of speaker and phonetic identity in the CPC representation space.

**Linear Probe** We trained two linear classifiers to predict the speaker and phone label given a single CPC vector. For each of the 40 speakers in the dev-clean subset of LibriSpeech, we randomly selected 10 utterances to form the test set and used the rest for training.

## 5 Results

As shown in Table 1, neither the standardisation nor the linear mapping operation makes much difference in ABX score. In fact, the highest score for both within- and across-ABX was achieved by

| Standard-isation | Linear Mapping | ABX score |  | Probing Acc. |  |
|---|---|---|---|---|---|
|  |  | Within | Across | Speaker | Phone |
|  |  | 96.62 | 95.87 | 99.48 | 76.00 |
| ✓ |  | 96.55 | 95.87 | 21.72 | 75.58 |
|  | ✓ | 96.62 | 94.36 | 4.62 | 72.33 |
| ✓ | ✓ | 96.55 | 94.51 | 7.89 | 71.83 |

Table 1: ABX score and probing accuracy for CPC representations after standardisation and/or linear mapping

the original CPC representations. Greater gaps were observed in the probing experiments, particularly in speaker classification. While speaker identity can be almost perfectly predicted given the original CPC representations, utterance-level standardisation could reduce the classification accuracy by 77% to 22%. Applying linear mapping computed from mean phone vectors would reduce the accuracy to less than 5%, although it comes with a larger drop in the probing accuracy for phone identity than standardisation. We leave it to future work to examine how much performance drop in downstream spoken language modeling would result from this loss in phone information.

In an additional experiment, we tried performing dimensionality reduction via PCA before the linear mapping operation. We found that, when reduced to 100 dimensions, the representations can give a probing accuracy of 2.34% for speaker identity while maintaining an accuracy of 70.51% for phone identity. Even when reduced to 10 dimensions, the CPC representations achieved higher ABX score than (within: 96.97%, across 86.65%) 39-dimensional MFCC features (within: 90.68%, across 79.89%).

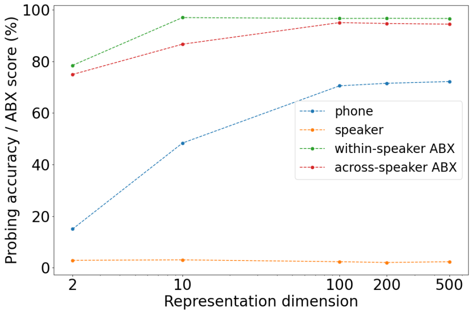


Figure 5: ABX score and probing accuracy for CPC representations after dimensionality reduction

## 6 Future Directions

In this project, we showed that speaker information in CPC representations can be removed to a large extent by aligning the mean phone vectors between speakers. However, this operation also causes loss in phonetic information, and we will continue to investigate the effect of this on downstream spoken language modeling tasks.

Despite causing marked changes in probing accuracy, our proposed method resulted in very little difference in ABX score, which could be because the ABX test set we used is too easy for the model. It was extracted from read speech in audiobooks, which tends to be much less challenging than spontaneous speech in conversation recordings. Also, the CPC model we used was trained on data coming from the same LibriSpeech dataset, albeit different subsets, which implies there was little domain shift. Therefore, testing on a different dataset that is more challenging might allow us to see greater difference in ABX score.

We might also want to experiment with a different formulation of ABX. For speaker-invariant representations, we hope to see that the similarity between the same phone produced by two different speakers should be higher than that between two different phones produced by the same speaker. However, this is not what the across-speaker ABX is designed to test for. Recall that for across-speaker ABX, there is A = $\text{beg}_{S1}$, B = $\text{bag}_{S1}$, X = $\text{bag}_{S2}$. However, to test for speaker-invariance as we envisaged, the triplet should be set up as A = $\text{beg}_{S2}$, B = $\text{bag}_{S1}$, X = $\text{bag}_{S2}$.

It is important to think about ways of applying this post-processing method without supervision, since CPC is developed in the context of unsupervised learning after all. While the mean phone vectors we used to compute the alignment mapping were aggregated from all CPC vectors grouped by phone labels, there are two potential ways of obtaining proxies of the phone space in an unsupervised setting. First, the top eigenvectors of the matrix concatenated from all CPC representations of one speaker could capture some information about the phone space of that speaker. We carried out a preliminary analysis in the correlation between the top eigenvectors and the mean phone vectors of the same speaker. We found that the top first eigenvector distinguishes vowels and consonants, with mean vectors for vowels pointing in the same direction (*i.e.* positive cosine similarity) as the eigenvector and those for consonants in the opposite direction. The second top eigenvector discriminates fricatives from others, and the third distinguishes nasal sounds. However, there is the ambiguity in defining the directionality of each eigenvector, namely that the vector resulted from flipping the direction of an eigenvector is also an eigenvector. Moreover, beyond the top few eigenvectors, the phone information captured by each eigenvector gets noisier and the correspondence across speakers is lost. Nevertheless, our next experiment will focus on using eigenvectors of each speaker for alignment in a similar manner to Fernando et al. (2014). Another option is to use representations of the same word or utterance token to align two speakers. Our first step in this direction would be to use the disclaimer utterance produced by each speaker in the LibriSpeech dataset to align them. These utterances have the same content but may differ in duration, hence again dynamic time warping needs to be applied to representation sequences before computing the linear mapping.

is there a better way of putting this?

The problem of different ordering that we talked about in a previous meeting. Like for speaker 1, the 10th eigenvector distinguishes nasals but for speaker 2, the 12th eigenvector does that. Not sure how to concisely express this.

## References

María Andrea Cruz Blandón and Okko Räsänen. 2020. Analysis of predictive coding models for phonemic representation learning in small datasets. *arXiv preprint arXiv:2007.04205*.

Yu-An Chung, Wei-Ning Hsu, Hao Tang, and James Glass. 2019. An unsupervised autoregressive model for speech representation learning. *arXiv preprint arXiv:1904.03240*.

Naomi H Feldman, Sharon Goldwater, Emmanuel Dupoux, and Thomas Schatz. 2021. Do infants really learn phonetic categories? *Open Mind*, pages 1–19.

Basura Fernando, Amaury Habrard, Marc Sebban, and Tinne Tuytelaars. 2014. Subspace alignment for domain adaptation. *arXiv preprint arXiv:1409.5241*.

Jacob Kahn, Morgane Riviere, Weiyi Zheng, Evgeny Kharitonov, Qiantong Xu, Pierre-Emmanuel Mazaré, Julien Karadayi, Vitaliy Liptchinsky, Ronan Collobert, Christian Fuegen, et al. 2020. Libri-light: A benchmark for asr with limited or no supervision. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7669–7673. IEEE.

Patricia K Kuhl, Erica Stevens, Akiko Hayashi, Toshisada Deguchi, Shigeru Kiritani, and Paul Iverson. 2006. Infants show a facilitation effect for native language phonetic perception between 6 and 12 months. *Developmental science*, 9(2):F13–F21.

Andy T Liu, Shu-wen Yang, Po-Han Chi, Po-chun Hsu, and Hung-yi Lee. 2020. Mockingjay: Unsupervised speech representation learning with deep bidirectional transformer encoders. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6419–6423. IEEE.

Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Tu Anh Nguyen, Maureen de Seyssel, Patricia Rozé, Morgane Rivière, Evgeny Kharitonov, Alexei Baevski, Ewan Dunbar, and Emmanuel Dupoux. 2020. The zero resource speech benchmark 2021: Metrics and baselines for unsupervised spoken language modeling.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2019. Representation learning with contrastive predictive coding.

Nikolay Oskolkov. 2021. Tsne vs. umap: Global structure.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE.

Morgane Riviere, Armand Joulin, Pierre-Emmanuel Mazaré, and Emmanuel Dupoux. 2020. Unsupervised pretraining transfers well across languages. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7414–7418. IEEE.

Thomas Schatz, Vijayaditya Peddinti, Francis Bach, Aren Jansen, Hynek Hermansky, and Emmanuel Dupoux. 2013. Evaluating speech features with the minimal-pair abx task: Analysis of the classical mfc/plp pipeline. In *INTERSPEECH 2013: 14th Annual Conference of the International Speech Communication Association*, pages 1–5.

Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. 2019. wav2vec: Unsupervised pre-training for speech recognition. *arXiv preprint arXiv:1904.05862*.

Benjamin van Niekerk, Leanne Nortje, Matthew Baas, and Herman Kamper. 2021. Analyzing Speaker Information in Self-Supervised Models to Improve Zero-Resource Speech Processing. *arXiv:2108.00917 [cs, eess]*.