Deep convolutional acoustic word embeddings using word-pair side information

Herman Kamper[1], Weiran Wang[2], Karen Livescu[2]

[1]CSTR and ILCC, School of Informatics, University of Edinburgh, UK
[2]Toyota Technological Institute at Chicago, USA

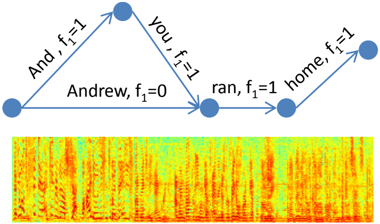
ICASSP 2016

Introduction

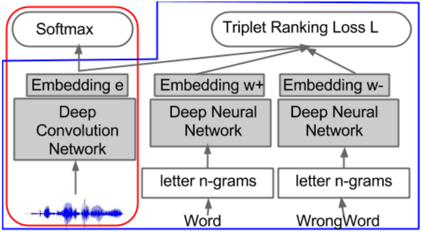
- Most speech processing systems rely on deep architecture to classify speech frames into subword units (HMM triphone states).
- Requires pronunciation dictionary for breaking words into subwords; in many cases still make frame-level independence assumptions.
- Some studies have started to reconsider whole words as basic modelling unit [Heigold *et al.*, 2012; Chen *et al.*, 2015].

Segmental automatic speech recognition

Segmental conditional random field ASR [Maas *et al.*, 2012]:



Whole-word lattice rescoring [Bengio and Heigold, 2014]:



Segmental query-by-example search

From [Levin *et al.*, 2015]:

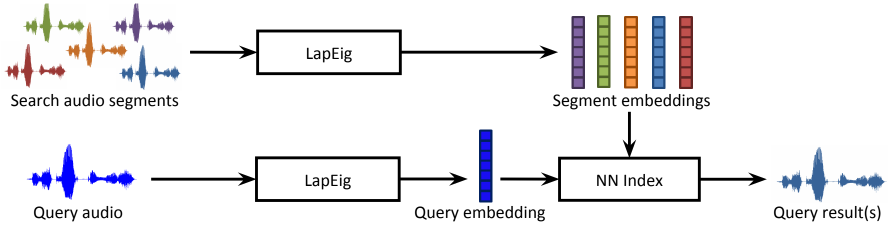


Fig. 1. Diagram of the S-RAILS audio search system.

[Chen *et al.*, 2015]: Similar scheme for "Okay Google" using LSTMs.

Segmental query-by-example search

From [Levin *et al.*, 2015]:

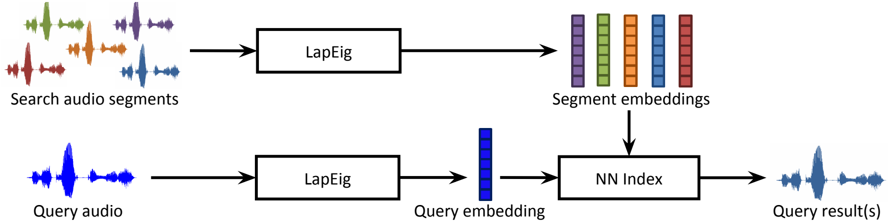
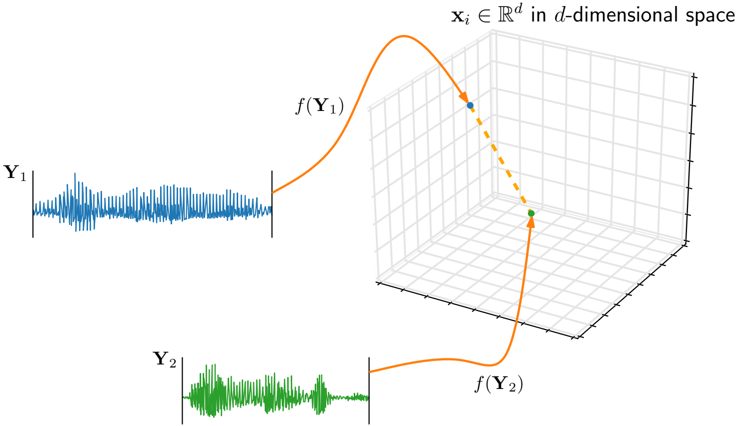


Fig. 1. Diagram of the S-RAILS audio search system.

[Chen *et al.*, 2015]: Similar scheme for "Okay Google" using LSTMs.

In this work, we also use a query-related task for evaluation.

Acoustic word embedding problem



Reference vector method [Levin *et al.*, 2013]

Reference vector method [Levin *et al.*, 2013]

Segment we want to embed:

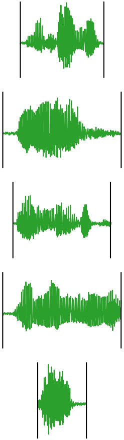
$\mathbf{y}_{t_1:t_2}$

Reference vector method [Levin *et al.*, 2013]

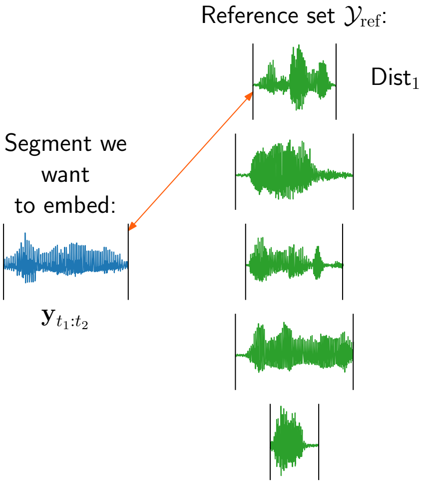
Segment we want to embed:

$\mathbf{y}_{t_1:t_2}$

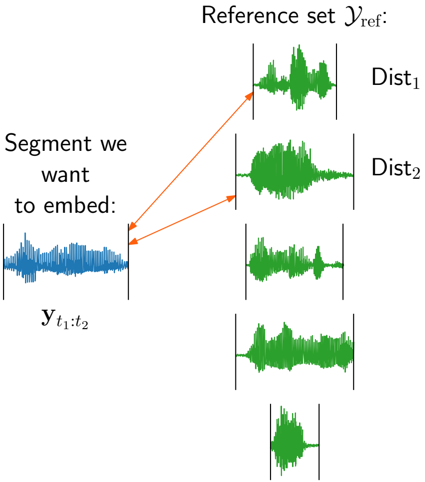
Reference set $\mathcal{Y}_{\text{ref}}$:



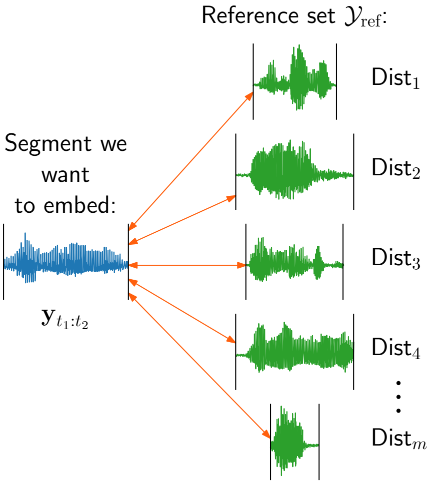
Reference vector method [Levin *et al.*, 2013]



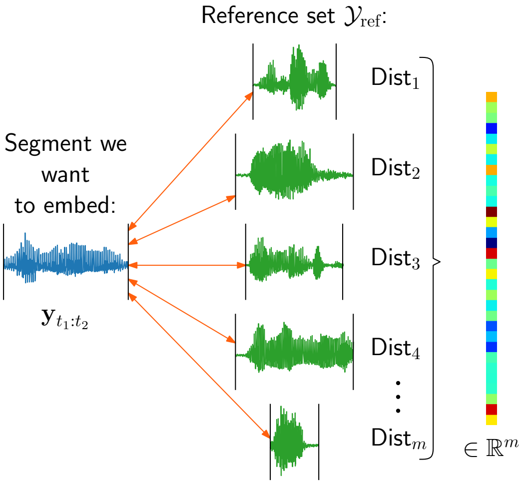
Reference vector method [Levin *et al.*, 2013]



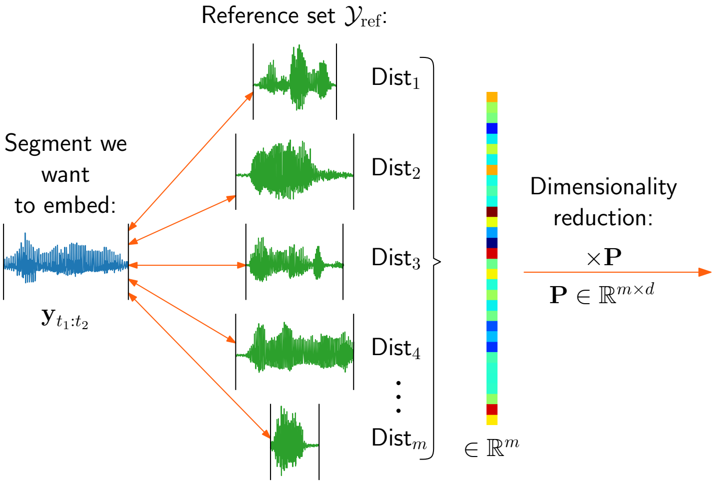
Reference vector method [Levin *et al.*, 2013]



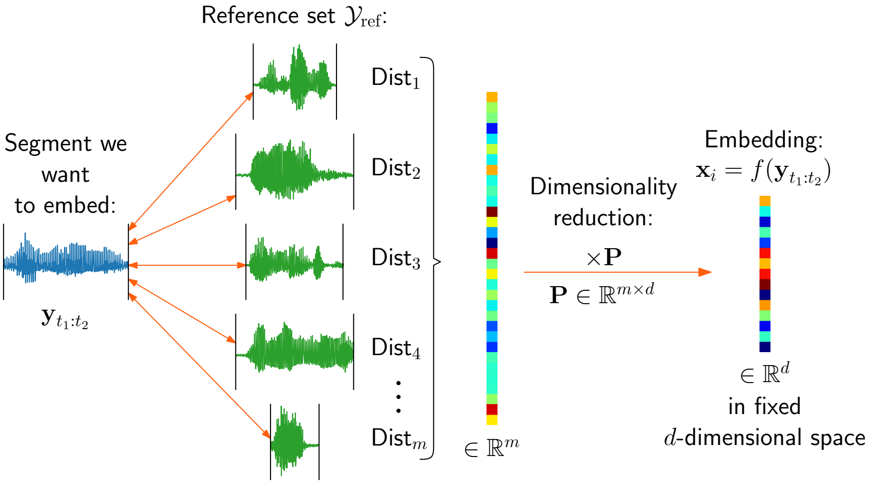
Reference vector method [Levin *et al.*, 2013]



Reference vector method [Levin *et al.*, 2013]



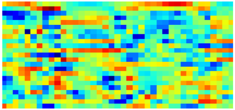
Reference vector method [Levin *et al.*, 2013]



Word classification CNN [Bengio and Heigold, 2014]

Word classification CNN [Bengio and Heigold, 2014]

$$\overset{w_i}{\boxed{0\ 0\ 0\ \cdots\ 1\ \cdots\ 0\ 0}}$$



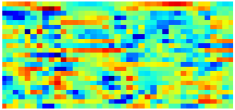
$\mathbf{Y}_i$

Word classification CNN [Bengio and Heigold, 2014]

softmax

w_i

$\boxed{0\ 0\ 0\ \cdots\ 1\ \cdots\ 0\ 0}$



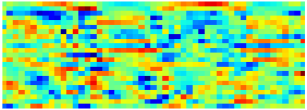
$\mathbf{Y}_i$

Word classification CNN [Bengio and Heigold, 2014]

softmax

w_i

$\boxed{0\ 0\ 0\ \cdots\ 1\ \cdots\ 0\ 0}$



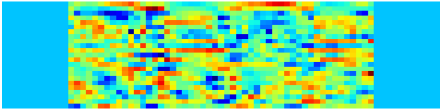
$\mathbf{Y}_i$

Word classification CNN [Bengio and Heigold, 2014]

softmax

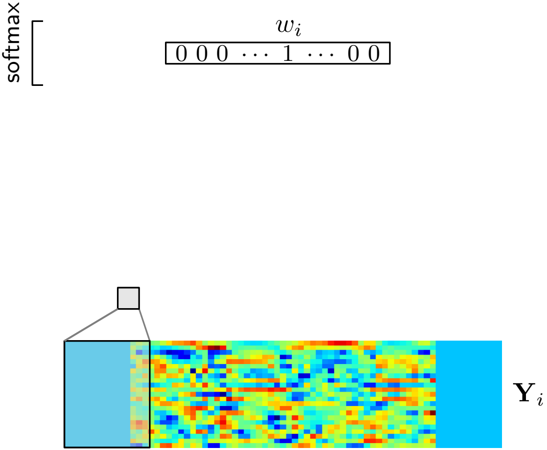
$$w_i$$

$$\boxed{0\ 0\ 0\ \cdots\ 1\ \cdots\ 0\ 0}$$

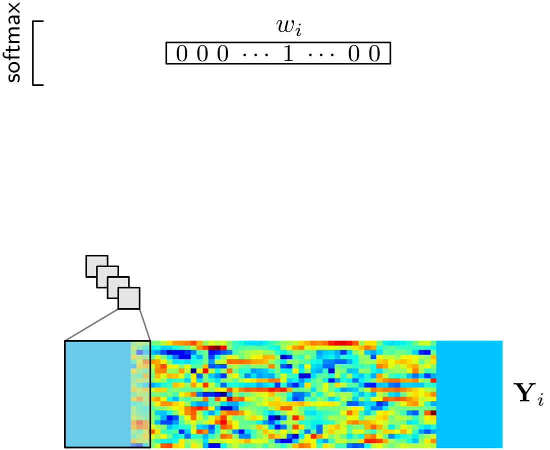


$\mathbf{Y}_i$

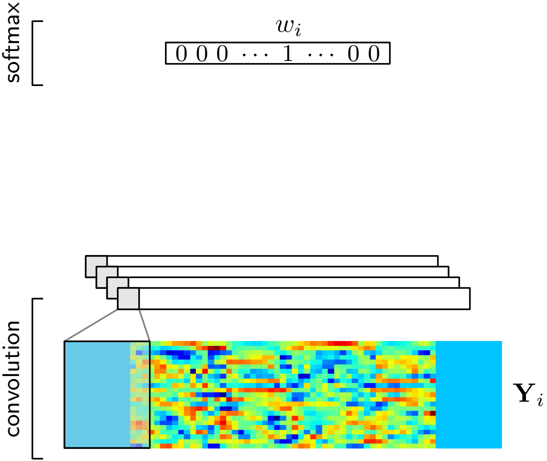
Word classification CNN [Bengio and Heigold, 2014]



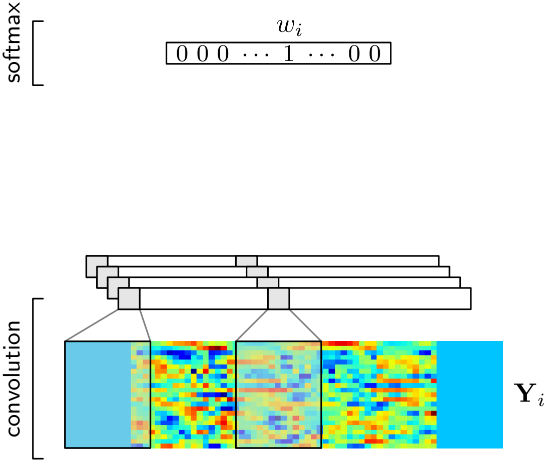
Word classification CNN [Bengio and Heigold, 2014]



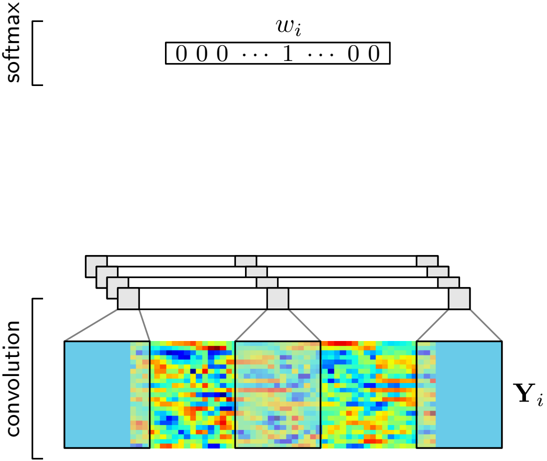
Word classification CNN [Bengio and Heigold, 2014]



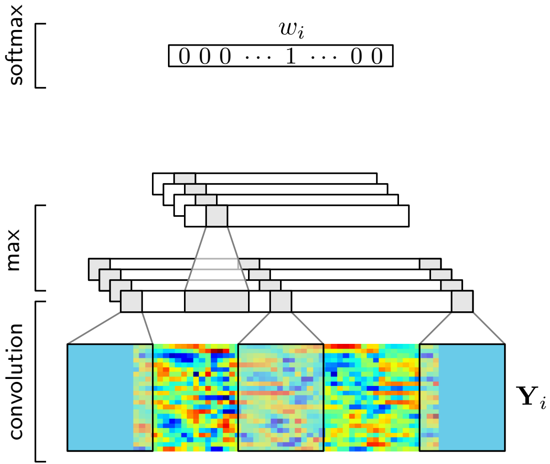
Word classification CNN [Bengio and Heigold, 2014]



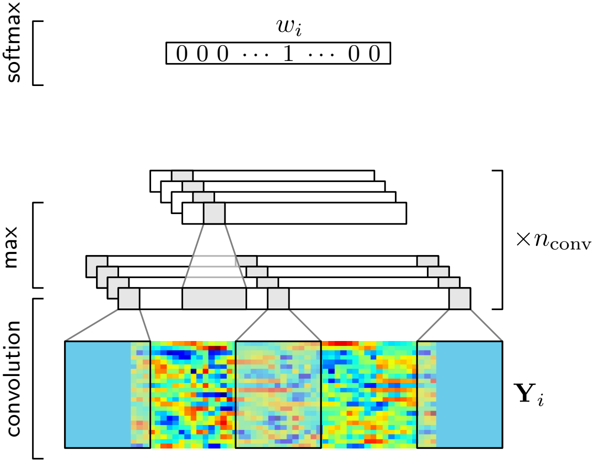
Word classification CNN [Bengio and Heigold, 2014]



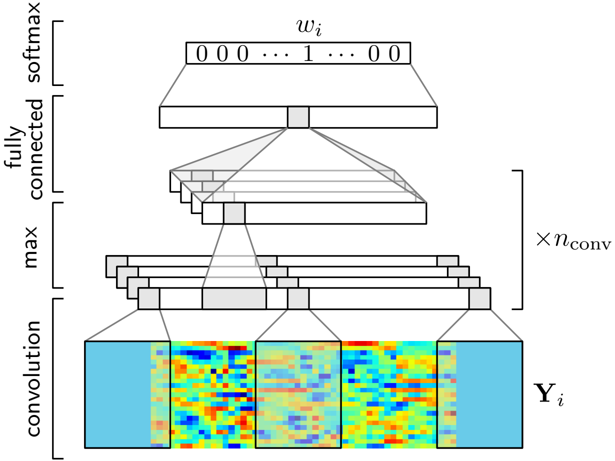
Word classification CNN [Bengio and Heigold, 2014]



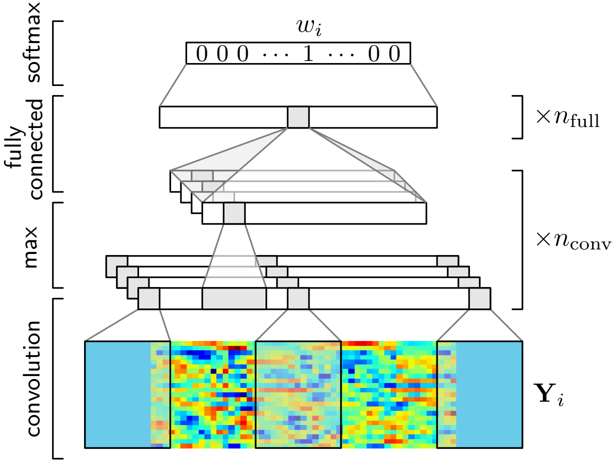
Word classification CNN [Bengio and Heigold, 2014]



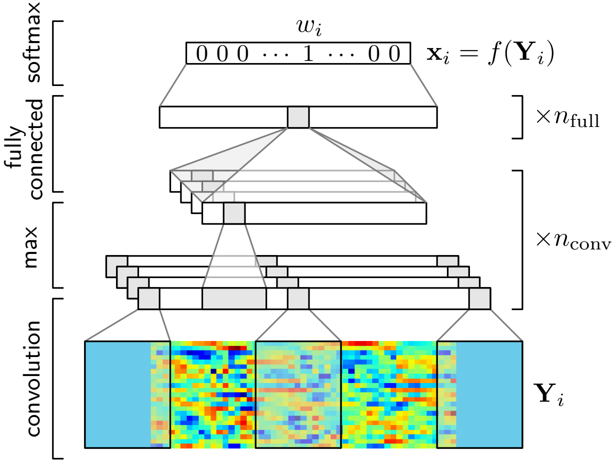
Word classification CNN [Bengio and Heigold, 2014]



Word classification CNN [Bengio and Heigold, 2014]



Word classification CNN [Bengio and Heigold, 2014]



Supervision and side information

- The word classifier CNN assumes a corpus of labelled word segments.
- In some cases these might not be available.
- Weaker form of supervision we sometimes have (e.g. [Thiollière *et al.*, 2015]) are known word pairs: $\mathcal{S}_{\text{train}} = \{(m, n) : (Y_m, Y_n) \text{ are of the same type}\}$
- Also aligns with query / word discrimination task: does two speech segments contain instances of the same word? (Don't care about word identity.)

Supervision and side information

- The word classifier CNN assumes a corpus of labelled word segments.
- In some cases these might not be available.
- Weaker form of supervision we sometimes have (e.g. [Thiollière *et al.*, 2015]) are known word pairs: $\mathcal{S}_{\text{train}} = \{(m, n) : (Y_m, Y_n) \text{ are of the same type}\}$
- Also aligns with query / word discrimination task: does two speech segments contain instances of the same word? (Don't care about word identity.)

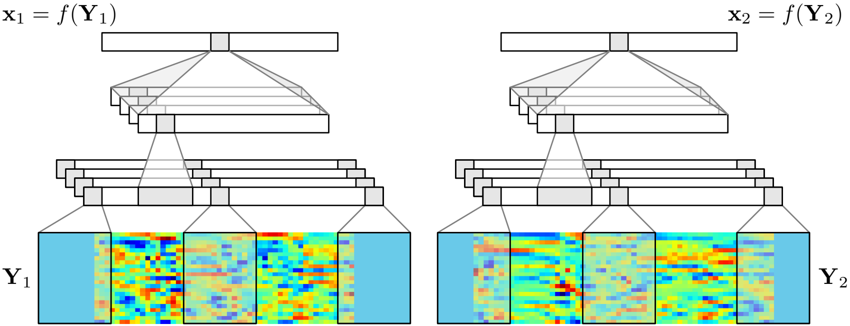
Can we use this weak supervision (sometimes called side information) to train an acoustic word embedding function f?

Word similarity Siamese CNN

Use idea of *Siamese networks* [Bromley *et al.*, 1993].

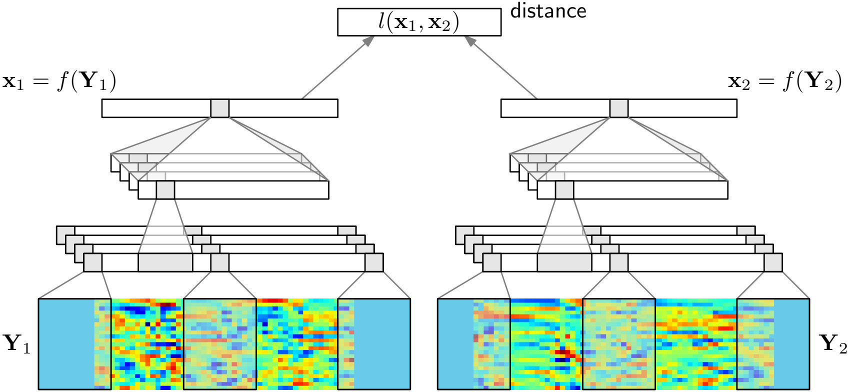
Word similarity Siamese CNN

Use idea of *Siamese networks* [Bromley *et al.*, 1993].



Word similarity Siamese CNN

Use idea of *Siamese networks* [Bromley *et al.*, 1993].

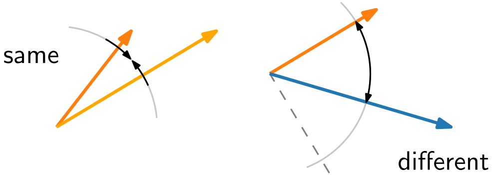


Loss functions

Loss functions

The coscos^2 loss [Synnaeve *et al.*, 2014]:

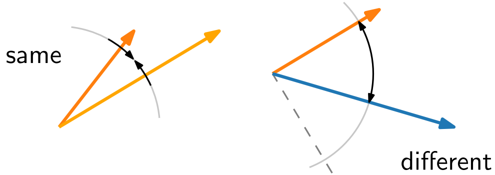
$$l_{\cos\ \cos^2}(\mathbf{x}_1, \mathbf{x}_2) = \begin{cases} \frac{1-\cos(\mathbf{x}_1,\mathbf{x}_2)}{2} & \text{if same} \\ \cos^2(\mathbf{x}_1, \mathbf{x}_2) & \text{if different} \end{cases}$$



Loss functions

The coscos^2 loss [Synnaeve *et al.*, 2014]:

$$l_{\cos\,\cos^2}(\mathbf{x}_1, \mathbf{x}_2) = \begin{cases} \frac{1-\cos(\mathbf{x}_1,\mathbf{x}_2)}{2} & \text{if same} \\ \cos^2(\mathbf{x}_1, \mathbf{x}_2) & \text{if different} \end{cases}$$



Margin-based hinge loss [Mikolov, 2013]:

$$l_{\cos\ \text{hinge}} = \max\left\{0, m + d_{\cos}(\mathbf{x}_1, \mathbf{x}_2) - d_{\cos}(\mathbf{x}_1, \mathbf{x}_3)\right\}$$

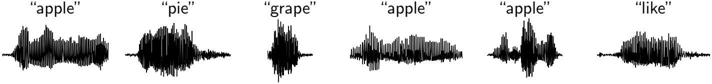
where $d_{\cos}(\mathbf{x}_1, \mathbf{x}_2) = \frac{1-\cos(\mathbf{x}_1,\mathbf{x}_2)}{2}$ is the cosine distance between $\mathbf{x}_1$ and $\mathbf{x}_2$, and m is a margin parameter. Pair $(\mathbf{x}_1, \mathbf{x}_2)$ are same, $(\mathbf{x}_1, \mathbf{x}_3)$ are different.

Embedding evaluation: the same-different task

Proposed in [Carlin *et al.*, 2011] and also used in [Levin *et al.*, 2013].

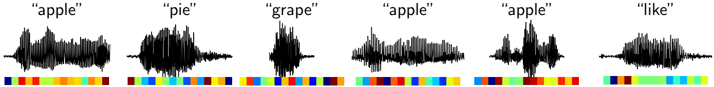
Embedding evaluation: the same-different task

Proposed in [Carlin *et al.*, 2011] and also used in [Levin *et al.*, 2013].



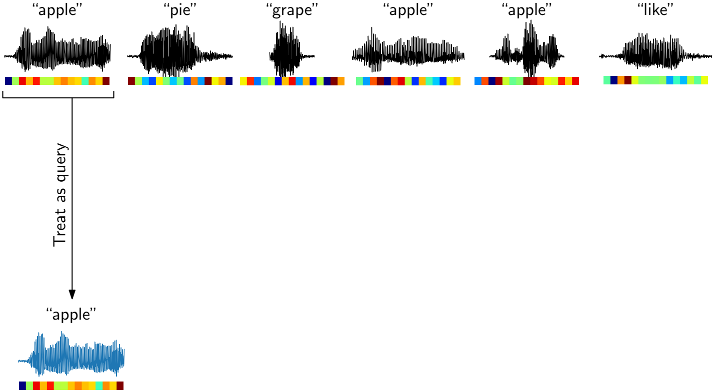
Embedding evaluation: the same-different task

Proposed in [Carlin *et al.*, 2011] and also used in [Levin *et al.*, 2013].



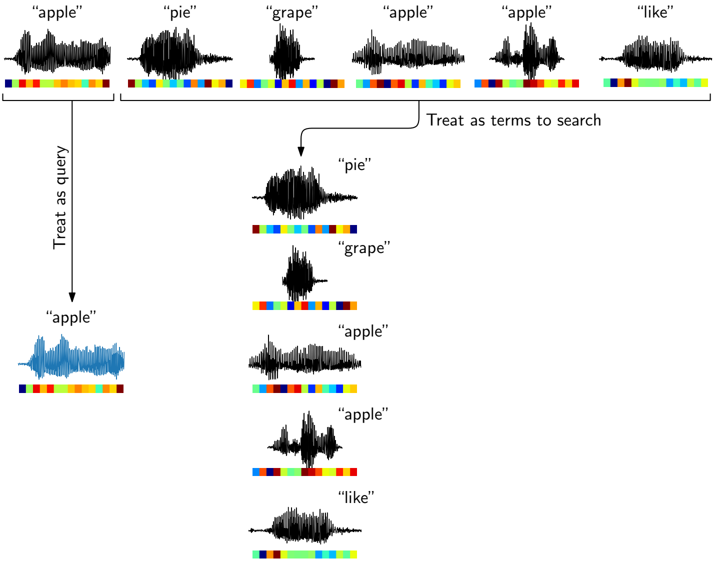
Embedding evaluation: the same-different task

Proposed in [Carlin *et al.*, 2011] and also used in [Levin *et al.*, 2013].



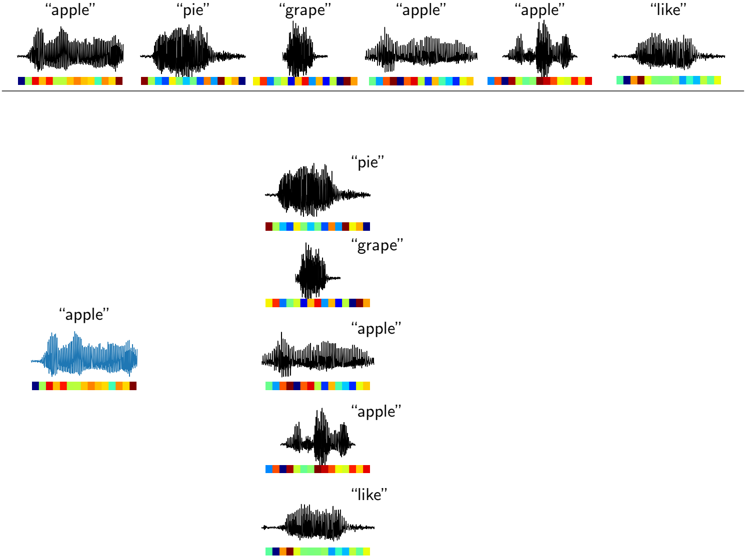
Embedding evaluation: the same-different task

Proposed in [Carlin *et al.*, 2011] and also used in [Levin *et al.*, 2013].



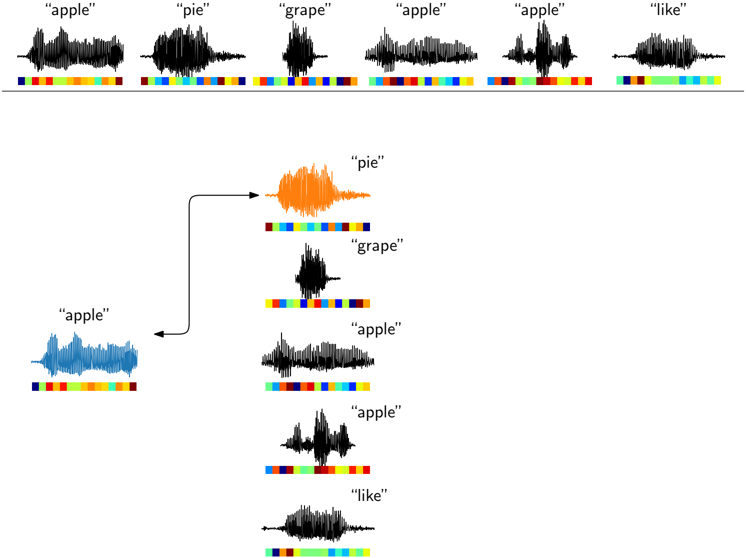
Embedding evaluation: the same-different task

Proposed in [Carlin *et al.*, 2011] and also used in [Levin *et al.*, 2013].



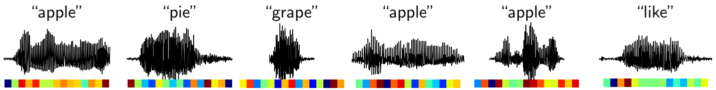
Embedding evaluation: the same-different task

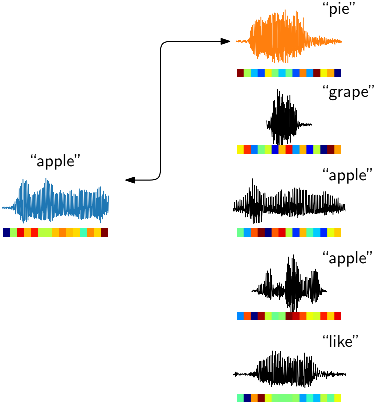
Proposed in [Carlin *et al.*, 2011] and also used in [Levin *et al.*, 2013].



Embedding evaluation: the same-different task

Proposed in [Carlin *et al.*, 2011] and also used in [Levin *et al.*, 2013].

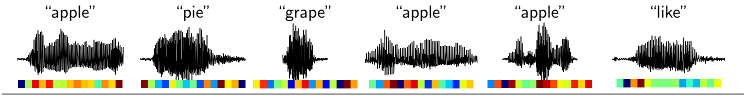


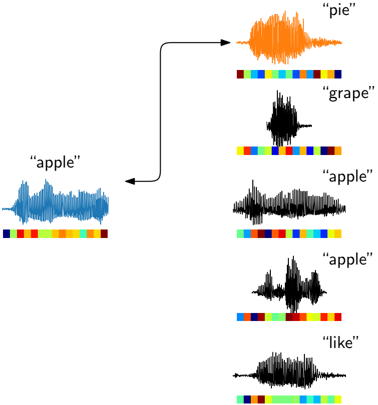


Cosine distance:	$d_i <$ threshold? predict:
d_1	

Embedding evaluation: the same-different task

Proposed in [Carlin *et al.*, 2011] and also used in [Levin *et al.*, 2013].

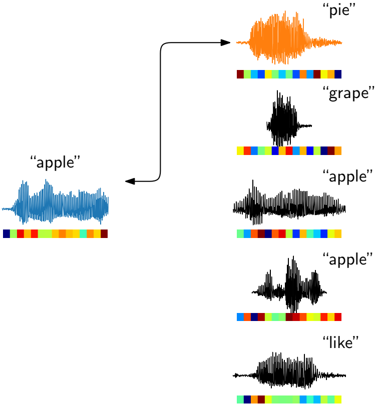




Cosine distance:	$d_i <$ threshold? predict:
d_1	different

Embedding evaluation: the same-different task

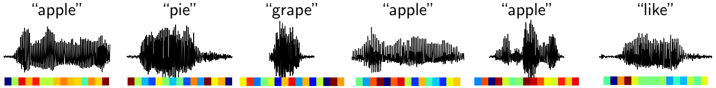
Proposed in [Carlin *et al.*, 2011] and also used in [Levin *et al.*, 2013].

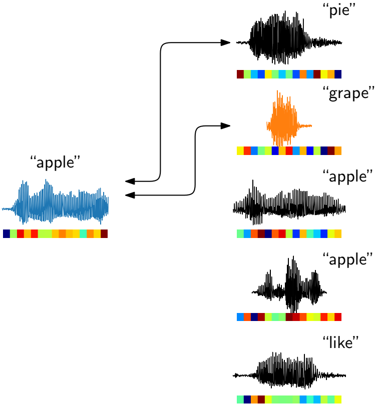


Cosine distance:	$d_i <$ threshold? predict:
d_1	different ✓

Embedding evaluation: the same-different task

Proposed in [Carlin *et al.*, 2011] and also used in [Levin *et al.*, 2013].

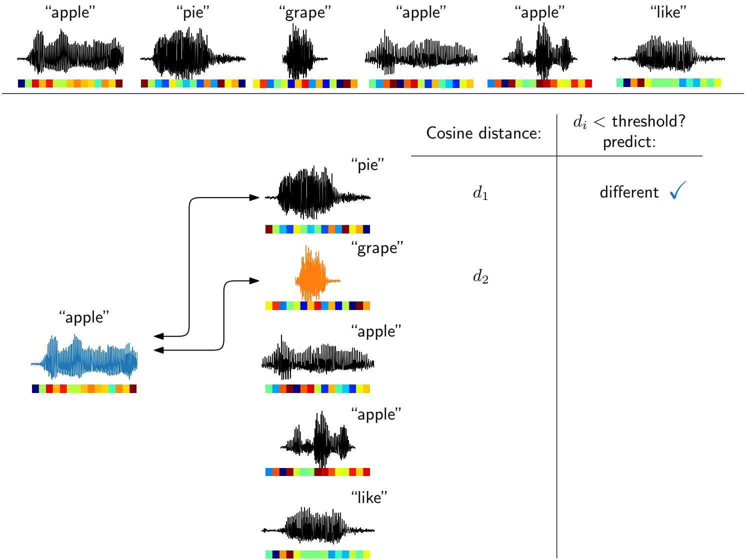




Cosine distance:	$d_i <$ threshold? predict:
d_1	different ✓

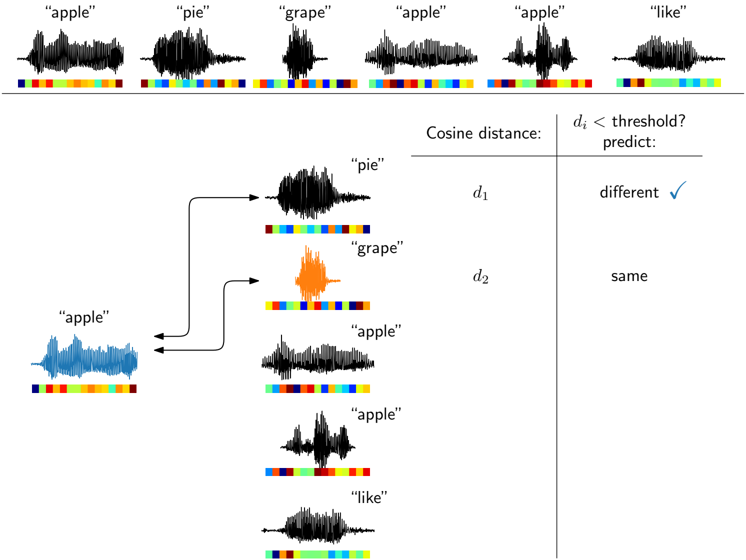
Embedding evaluation: the same-different task

Proposed in [Carlin *et al.*, 2011] and also used in [Levin *et al.*, 2013].



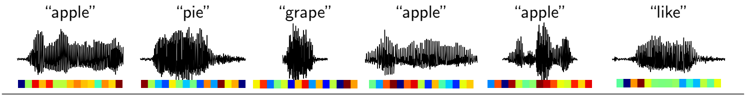
Embedding evaluation: the same-different task

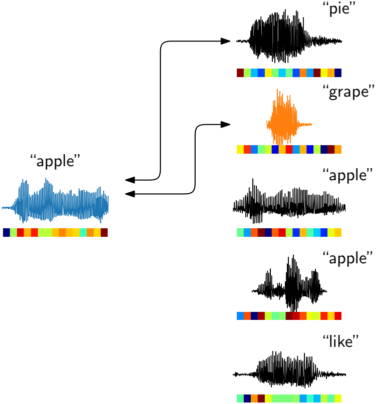
Proposed in [Carlin *et al.*, 2011] and also used in [Levin *et al.*, 2013].



Embedding evaluation: the same-different task

Proposed in [Carlin *et al.*, 2011] and also used in [Levin *et al.*, 2013].

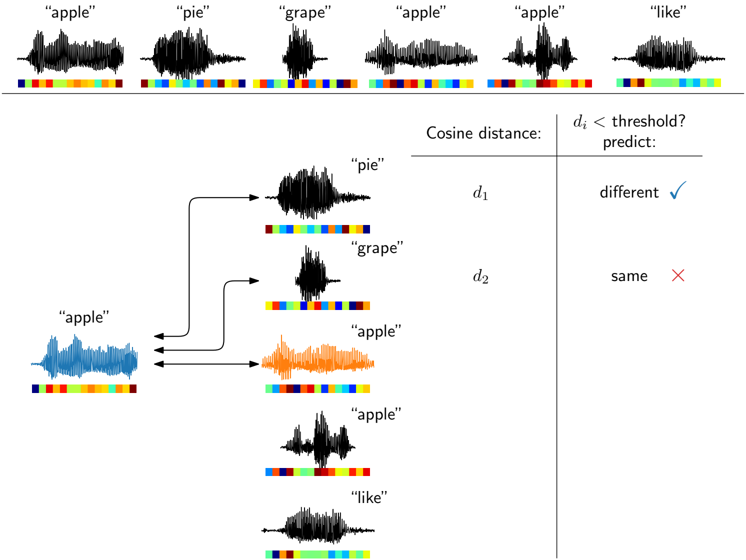




Cosine distance:	$d_i <$ threshold? predict:
d_1	different ✓
d_2	same ×

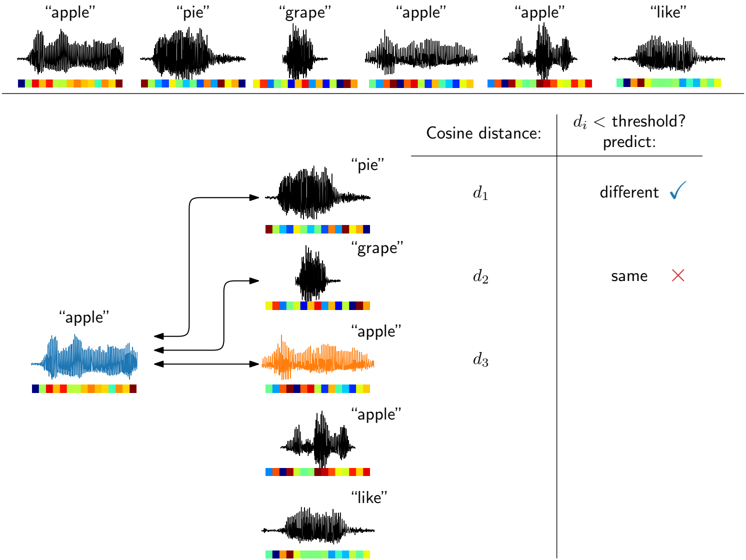
Embedding evaluation: the same-different task

Proposed in [Carlin *et al.*, 2011] and also used in [Levin *et al.*, 2013].



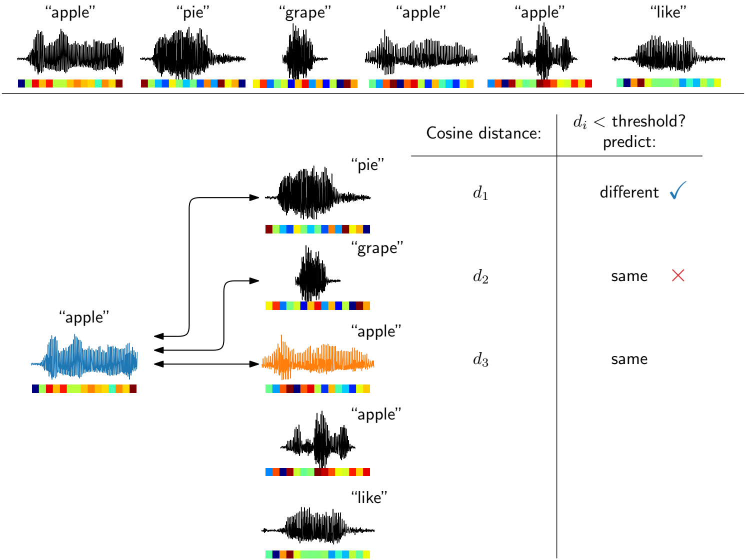
Embedding evaluation: the same-different task

Proposed in [Carlin *et al.*, 2011] and also used in [Levin *et al.*, 2013].



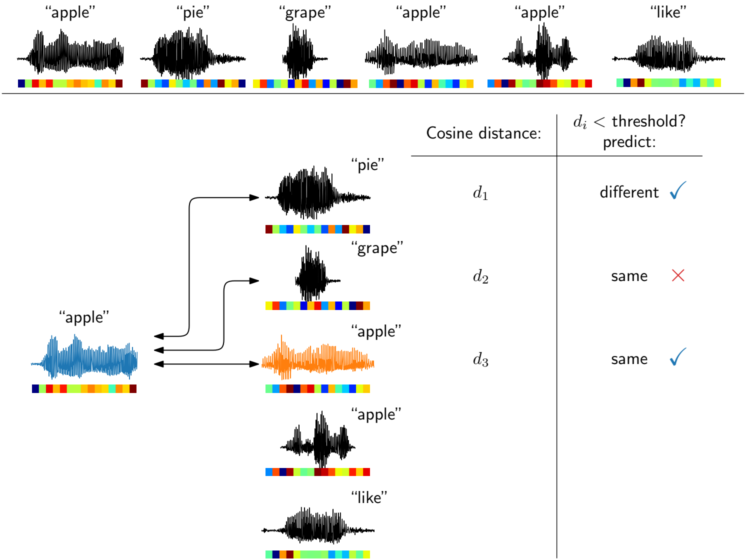
Embedding evaluation: the same-different task

Proposed in [Carlin *et al.*, 2011] and also used in [Levin *et al.*, 2013].



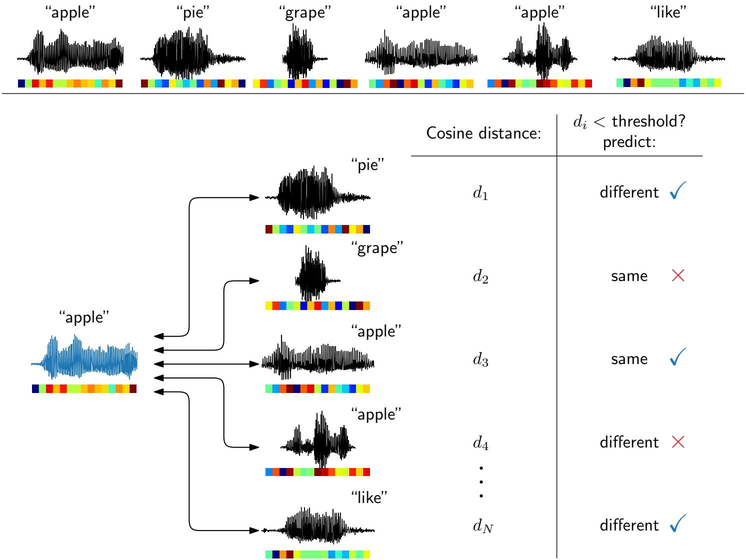
Embedding evaluation: the same-different task

Proposed in [Carlin *et al.*, 2011] and also used in [Levin *et al.*, 2013].



Embedding evaluation: the same-different task

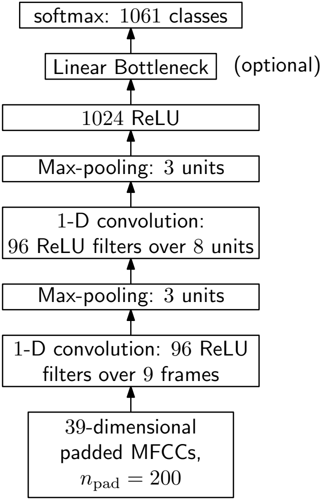
Proposed in [Carlin *et al.*, 2011] and also used in [Levin *et al.*, 2013].



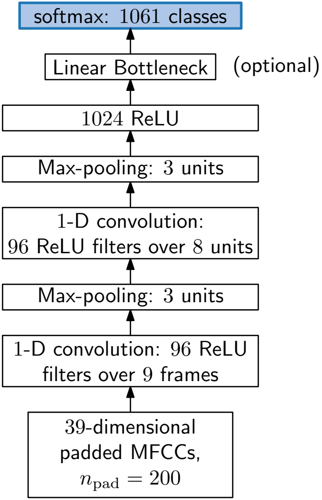
Experimental setup

- Speech from Switchboard is used for evaluation.
- Training set: 10k word tokens; sampled 100k training word pairs.
- Test set for same-different evaluation: 11k word tokens, 60.7M pairs, 3% produced by same speaker.
- Used a comparable development set.

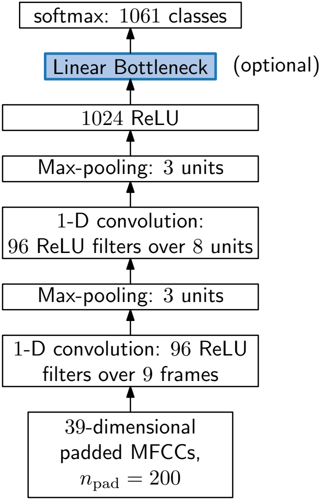
Network architectures: Word classifier CNN



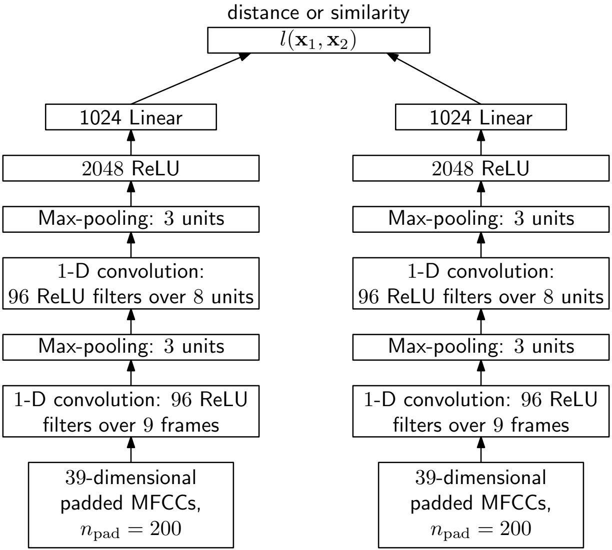
Network architectures: Word classifier CNN



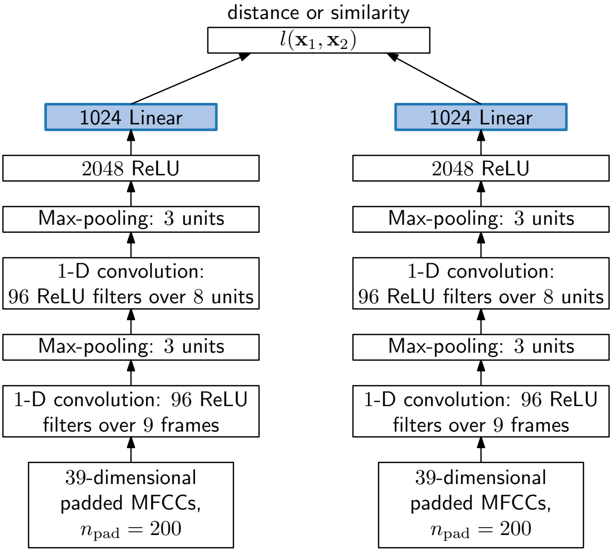
Network architectures: Word classifier CNN



Network architectures: Siamese CNN



Network architectures: Siamese CNN



Results

Representation	Dim	AP

Results

	Representation	Dim	AP
DTW	MFCCs with CMVN	39	0.214

Results

	Representation	Dim	AP
DTW	MFCCs with CMVN	39	0.214
DTW	Correspondence autoencoder [Kamper *et al.*, 2015]	100	0.469

Results

	Representation	Dim	AP
DTW	MFCCs with CMVN	39	0.214
	Correspondence autoencoder [Kamper *et al.*, 2015]	100	0.469
Acoustic word embed.	Reference vector approach [Levin *et al.*, 2013]	50	0.365

Results

	Representation	Dim	AP
DTW	MFCCs with CMVN	39	0.214
	Correspondence autoencoder [Kamper *et al.*, 2015]	100	0.469
Acoustic word embed.	Reference vector approach [Levin *et al.*, 2013]	50	0.365
	Word classifier CNN	1061	0.532 ± 0.014

Results

	Representation	Dim	AP
DTW	MFCCs with CMVN	39	0.214
DTW	Correspondence autoencoder [Kamper *et al.*, 2015]	100	0.469
Acoustic word embed.	Reference vector approach [Levin *et al.*, 2013]	50	0.365
Acoustic word embed.	Word classifier CNN	1061	0.532 ± 0.014
Acoustic word embed.		50	0.474 ± 0.012

Results

	Representation	Dim	AP
DTW	MFCCs with CMVN	39	0.214
	Correspondence autoencoder [Kamper *et al.*, 2015]	100	0.469
Acoustic word embed.	Reference vector approach [Levin *et al.*, 2013]	50	0.365
	Word classifier CNN	1061	0.532 ± 0.014
		50	0.474 ± 0.012
	Siamese CNN, $l_{\text{cos cos}^2}$ loss	1024	0.342 ± 0.026
	Siamese CNN, $l_{\text{cos hinge}}$ loss	1024	$\mathbf{0.549} \pm 0.011$

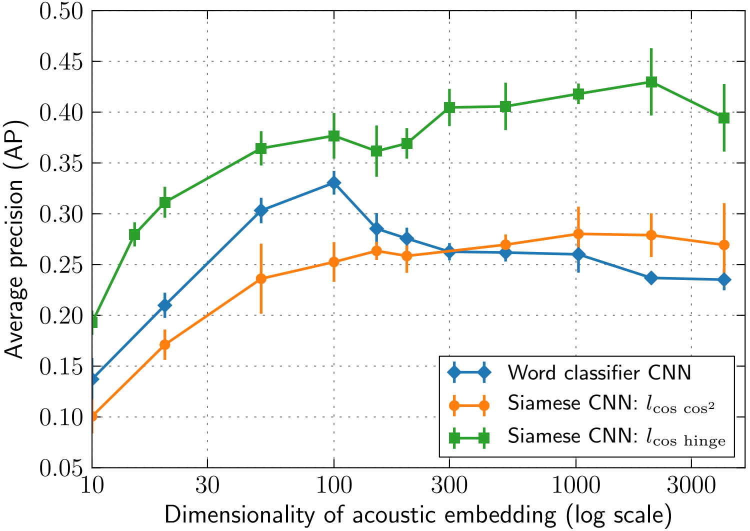
Results

	Representation	Dim	AP
DTW	MFCCs with CMVN	39	0.214
	Correspondence autoencoder [Kamper *et al.*, 2015]	100	0.469
Acoustic word embed.	Reference vector approach [Levin *et al.*, 2013]	50	0.365
	Word classifier CNN	1061	0.532 ± 0.014
		50	0.474 ± 0.012
	Siamese CNN, $l_{\text{cos cos}^2}$ loss	1024	0.342 ± 0.026
	Siamese CNN, $l_{\text{cos hinge}}$ loss	1024	$\mathbf{0.549} \pm 0.011$
		50	0.504 ± 0.011

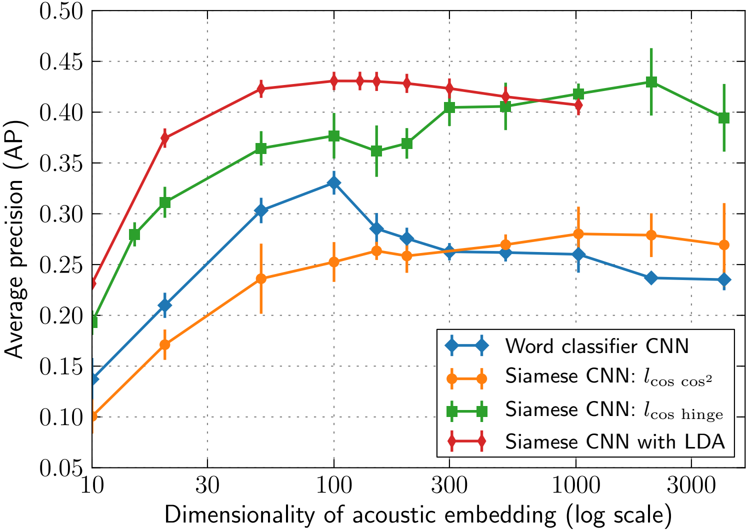
Results

	Representation	Dim	AP
DTW	MFCCs with CMVN	39	0.214
	Correspondence autoencoder [Kamper *et al.*, 2015]	100	0.469
Acoustic word embed.	Reference vector approach [Levin *et al.*, 2013]	50	0.365
	Word classifier CNN	1061	0.532 ± 0.014
		50	0.474 ± 0.012
	Siamese CNN, $l_{\text{cos cos}^2}$ loss	1024	0.342 ± 0.026
	Siamese CNN, $l_{\text{cos hinge}}$ loss	1024	$\mathbf{0.549} \pm 0.011$
		50	0.504 ± 0.011
	LDA on: $l_{\text{cos hinge}}$, $d = 1024$	100	0.545 ± 0.011

Varying dimensionalities on development data



Varying dimensionalities on development data



Summary and conclusion

- Introduced the Siamese CNN for obtaining acoustic word embeddings, and evaluated different cost functions.
- Evaluated using word discrimination task, and showed similar performance to word classifier CNN.
- For smaller dimensionalities: Siamese CNN outperformed classifier CNN.
- Self-criticism: evaluated on a small dataset (low-resource setting).
- Future work: sequence models, using embeddings for search and ASR.

Code

Neural networks (Theano): `https://github.com/kamperh/couscous`

Complete recipe: `https://github.com/kamperh/recipe_swbd_wordembeds`