Unsupervised Symbol Discovery

by applying language constraints on sequential data

Charly Lamothe [1] Thierry Artières [2] Ricard Marxer [3]

[1] Aix-Marseille Unversité
QARMA - éQuipe d'AppRentissage de MArseille

[2] ECM - École Centrale de Marseille
QARMA - éQuipe d'AppRentissage de MArseille
LIS - Laboratoire d'Informatique et Systèmes

[3] Université de Toulon
Aix-Marseille Unversité
LIS - Laboratoire d'Informatique et Systèmes

September 06, 2019

Summary

Introduction

Sequential data

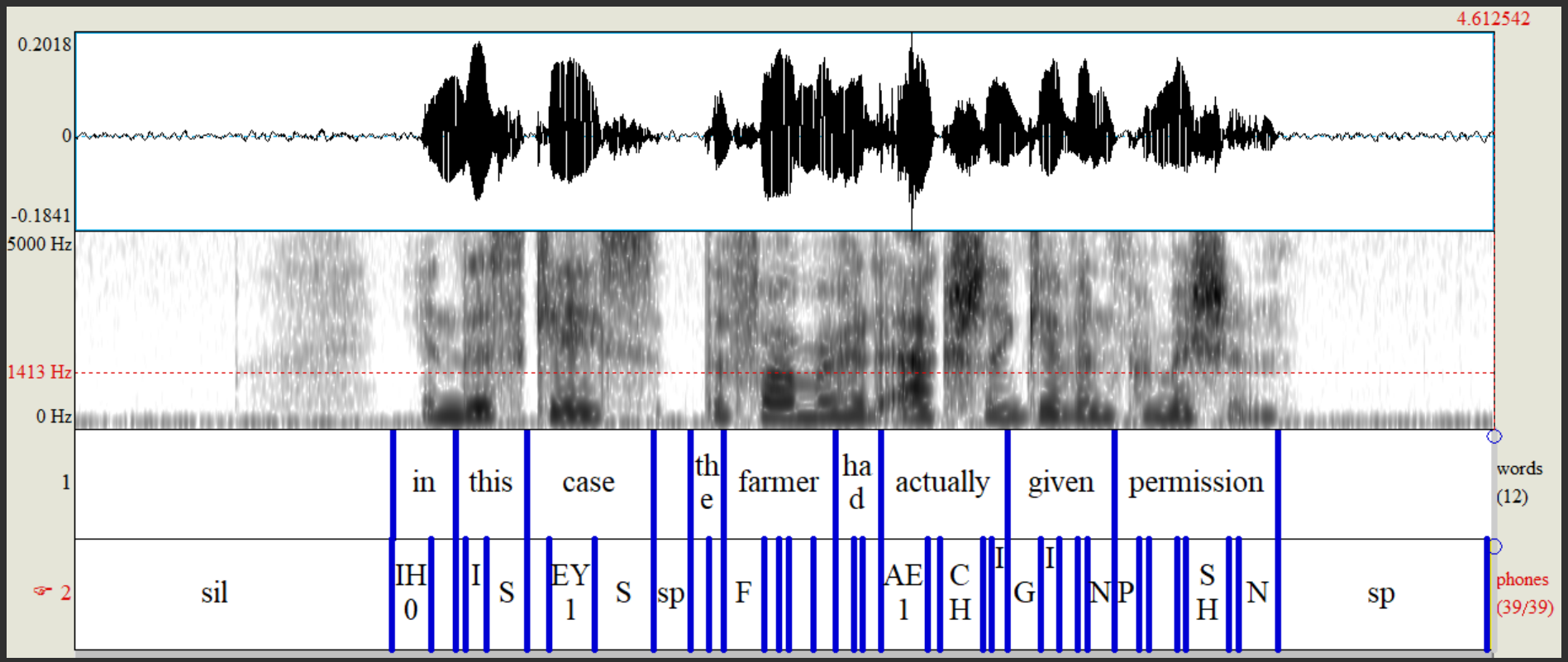


Figure: Example of speech signal with its aligned transcription.

Sequential data

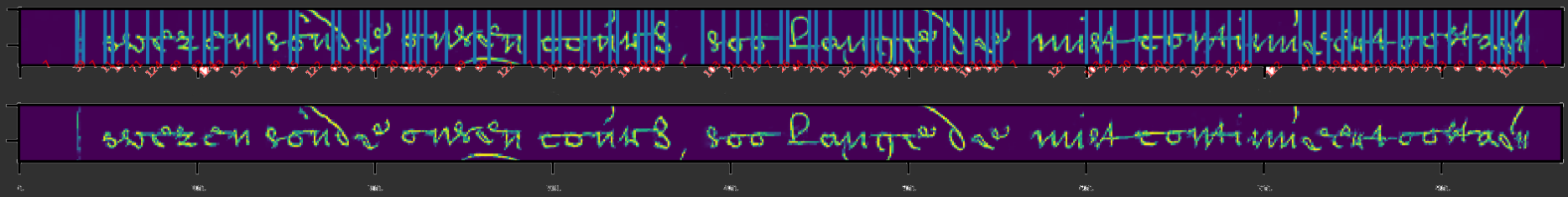
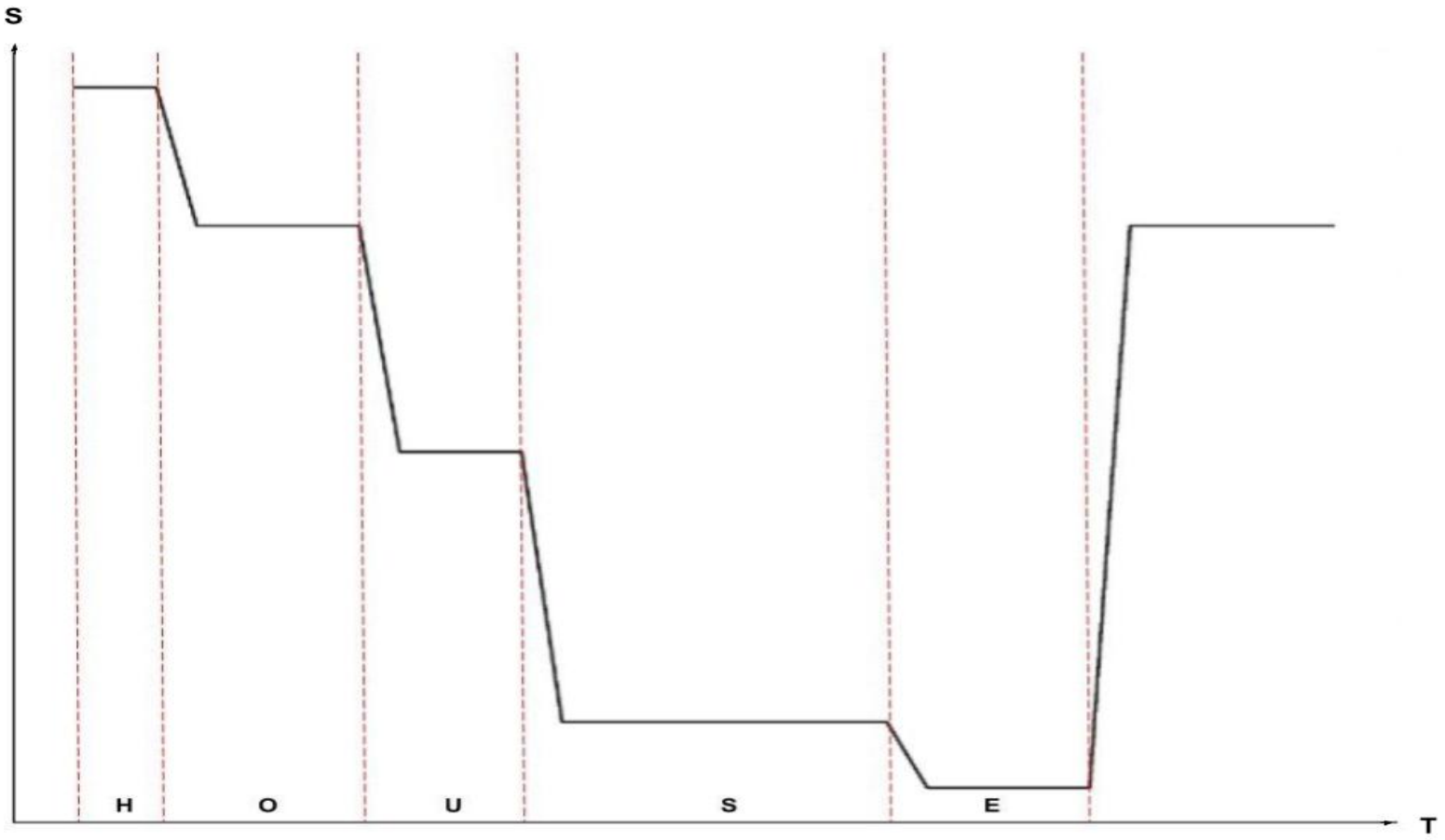
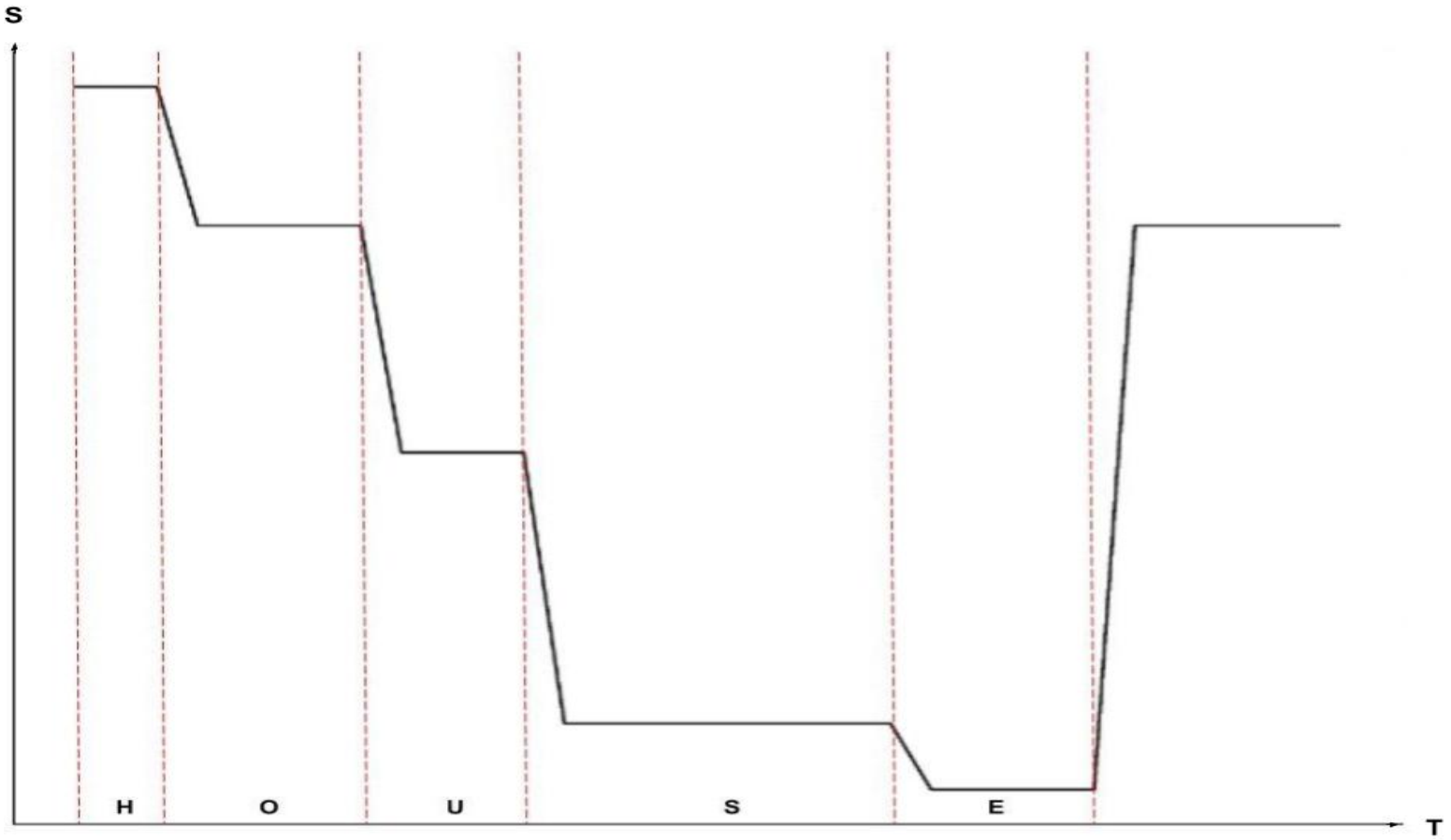


Figure: Example of handwritten text image with its aligned transcription.

Information in sequential structure



Information in sequential structure



$$S_{\text{house}} = -\sum_i P_i \log P_i \; \forall_i, P := \{P(H), P(O), P(U), P(S), P(E)\}$$

Hypotheses

(1) The existing statistical regularities of all languages $\mathbf{L}$ could be exploited to learn better representation of high level semantic content within sequential data;

(2) Different languages $\mathbf{L}$ share some equivalent statistical regularities;

(3) These regularities are shared across datasets $\mathcal{S}$ of different modalities $\mathbf{M}$.

Hypotheses

(1) The existing statistical regularities of all languages $\mathbf{L}$ could be exploited to learn better representation of high level semantic content within sequential data;

(2) Different languages $\mathbf{L}$ share some equivalent statistical regularities;

(3) These regularities are shared across datasets $\mathcal{S}$ of different modalities $\mathbf{M}$.

Hypotheses

(1) The existing statistical regularities of all languages $\mathbf{L}$ could be exploited to learn better representation of high level semantic content within sequential data;

(2) Different languages $\mathbf{L}$ share some equivalent statistical regularities;

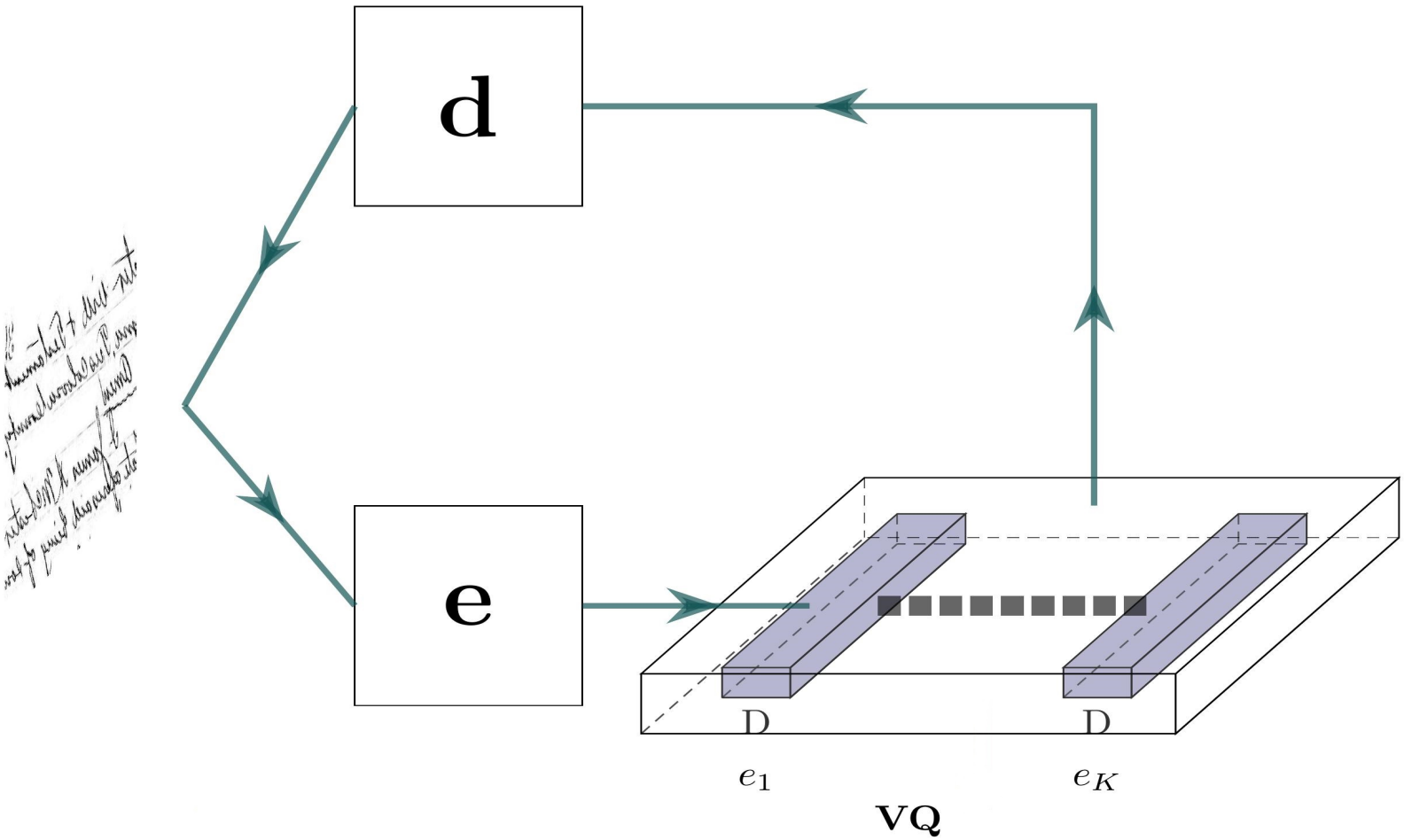
(3) These regularities are shared across datasets $\mathcal{S}$ of different modalities $\mathbf{M}$.

Preliminary notions

$\mathcal{S} := \{(x_i^{(t)}, y_i^{(t)})\}_{i=1}^{m}$ a dataset with $(x_i^{(t)}, y_i^{(t)}) \in \mathcal{X} \times \mathcal{Y}$;

$\forall\ y_i^{(t)} \in \mathcal{Y}$ an aligned transcription of $x_i^{(t)} \in \mathcal{X}$;

$\mathcal{S}_{\text{train}} \in \mathcal{S} \setminus \{y^{(t)}\}_{i=1}^{m},\ \mathcal{S}_{\text{eval}} \in \mathcal{S}$.



$\mathbf{e}$ encodes an input sequential data $x^{(t)}$ at timestep t such that $\hat{x}^{(t)} = \mathbf{d}(\mathbf{e}(x^{(t)}))$, where $z_{\mathbf{e}(x^{(t)})} = \mathbf{e}(x^{(t)})$ is a lower-dimensional space that keeps the most important features of $x^{(t)}$.

Vector Quantized-Variational Autoencoder (VQ-VAE)

Let $\{e_i \in \mathbb{R}^D, i = 1, \dots K\}$ be the K prototype vectors of dimension D;

$$q(x^{(t)}) = \text{argmin}_i ||z_{\text{e}(x^{(t)})} - e_i||_2^2 \; \forall i \in \{1, \dots, K\};$$

$$z_{q(x^{(t)})} = e_{q(x^{(t)})};$$

$$\hat{x}^{(t)} = \text{d}(z_{q(x^{(t)})});$$

$$\mathcal{L}_{\text{VQ-VAE}} = \mathcal{L}_{\text{err}} + ||\text{sg}(z_{\text{e}(x^{(t)})}) - e_{q(x^{(t)})}||_2^2 + \gamma ||z_{\text{e}(x^{(t)})} - \text{sg}(e_{q(x^{(t)})})||_2^2$$

Vector Quantized-Variational Autoencoder (VQ-VAE)

Let $\{e_i \in \mathbb{R}^D, i = 1, \dots K\}$ be the K prototype vectors of dimension D;

$q(x^{(t)}) = \text{argmin}_i ||z_{\mathbf{e}(x^{(t)})} - e_i||_2^2 \ \forall i \in \{1, \dots, K\}$;

$z_{q(x^{(t)})} = e_{q(x^{(t)})}$;

$\hat{x}^{(t)} = \mathbf{d}(z_{q(x^{(t)})})$;

$\mathcal{L}_{\text{VQ-VAE}} = \mathcal{L}_{\text{err}} + ||\text{sg}(z_{\mathbf{e}(x^{(t)})}) - e_{q(x^{(t)})}||_2^2 + \gamma ||z_{\mathbf{e}(x^{(t)})} - \text{sg}(e_{q(x^{(t)})})||_2^2$

Vector Quantized-Variational Autoencoder (VQ-VAE)

Let $\{e_i \in \mathbb{R}^D, i = 1, \dots K\}$ be the K prototype vectors of dimension D;

$q(x^{(t)}) = \text{argmin}_i ||z_{\mathbf{e}(x^{(t)})} - e_i||_2^2 \ \forall i \in \{1, \dots, K\}$;

$z_{q(x^{(t)})} = e_{q(x^{(t)})}$;

$\hat{x}^{(t)} = \mathbf{d}(z_{q(x^{(t)})})$;

$\mathcal{L}_{\text{VQ-VAE}} = \mathcal{L}_{\text{err}} + ||\text{sg}(z_{\mathbf{e}(x^{(t)})}) - e_{q(x^{(t)})}||_2^2 + \gamma ||z_{\mathbf{e}(x^{(t)})} - \text{sg}(e_{q(x^{(t)})})||_2^2$

Vector Quantized-Variational Autoencoder (VQ-VAE)

Let $\{e_i \in \mathbb{R}^D, i = 1, \dots K\}$ be the K prototype vectors of dimension D;

$q(x^{(t)}) = \text{argmin}_i ||z_{\mathbf{e}(x^{(t)})} - e_i||_2^2 \ \forall i \in \{1, \dots, K\}$;

$z_{q(x^{(t)})} = e_{q(x^{(t)})}$;

$\hat{x}^{(t)} = \mathbf{d}(z_{q(x^{(t)})})$;

$\mathcal{L}_{\text{VQ-VAE}} = \mathcal{L}_{\text{err}} + ||\text{sg}(z_{\mathbf{e}(x^{(t)})}) - e_{q(x^{(t)})}||_2^2 + \gamma ||z_{\mathbf{e}(x^{(t)})} - \text{sg}(e_{q(x^{(t)})})||_2^2$

Vector Quantized-Variational Autoencoder (VQ-VAE)

Let $\{e_i \in \mathbb{R}^D, i = 1, \dots K\}$ be the K prototype vectors of dimension D;

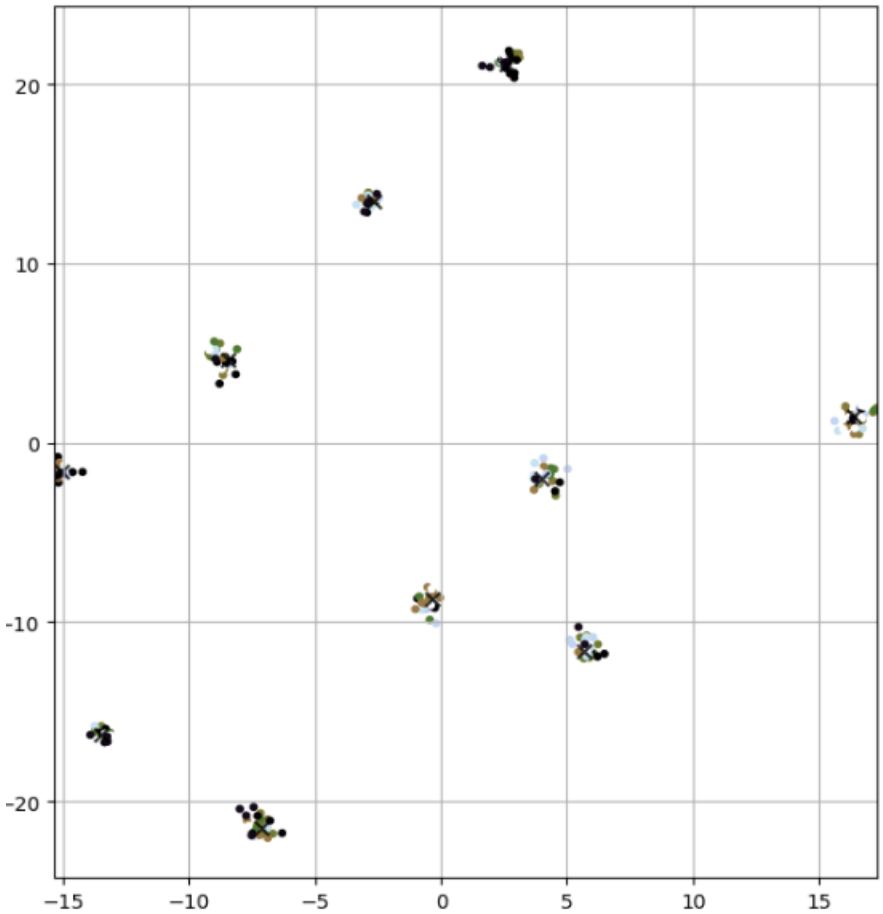
$q(x^{(t)}) = \text{argmin}_i ||z_{\mathbf{e}(x^{(t)})} - e_i||_2^2 \ \forall i \in \{1, \dots, K\}$;

$z_{q(x^{(t)})} = e_{q(x^{(t)})}$;

$\hat{x}^{(t)} = \mathbf{d}(z_{q(x^{(t)})})$;

$\mathcal{L}_{\text{VQ-VAE}} = \mathcal{L}_{\text{err}} + ||\text{sg}(z_{\mathbf{e}(x^{(t)})}) - e_{q(x^{(t)})}||_2^2 + \gamma ||z_{\mathbf{e}(x^{(t)})} - \text{sg}(e_{q(x^{(t)})})||_2^2$

Example of quantized embedding space



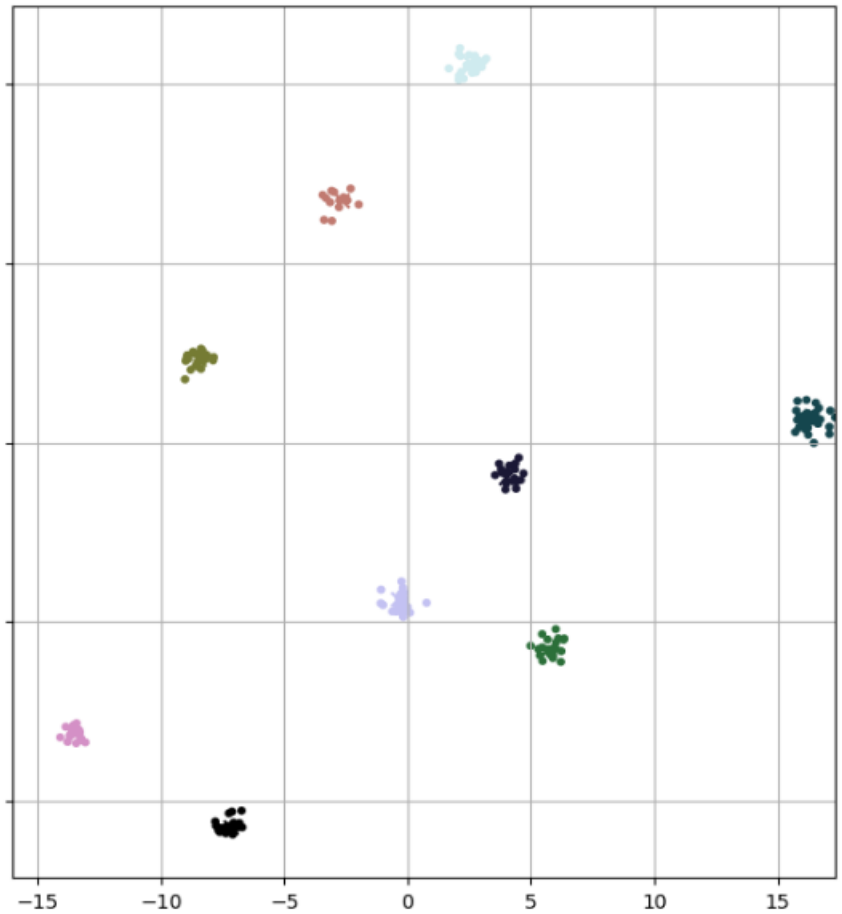
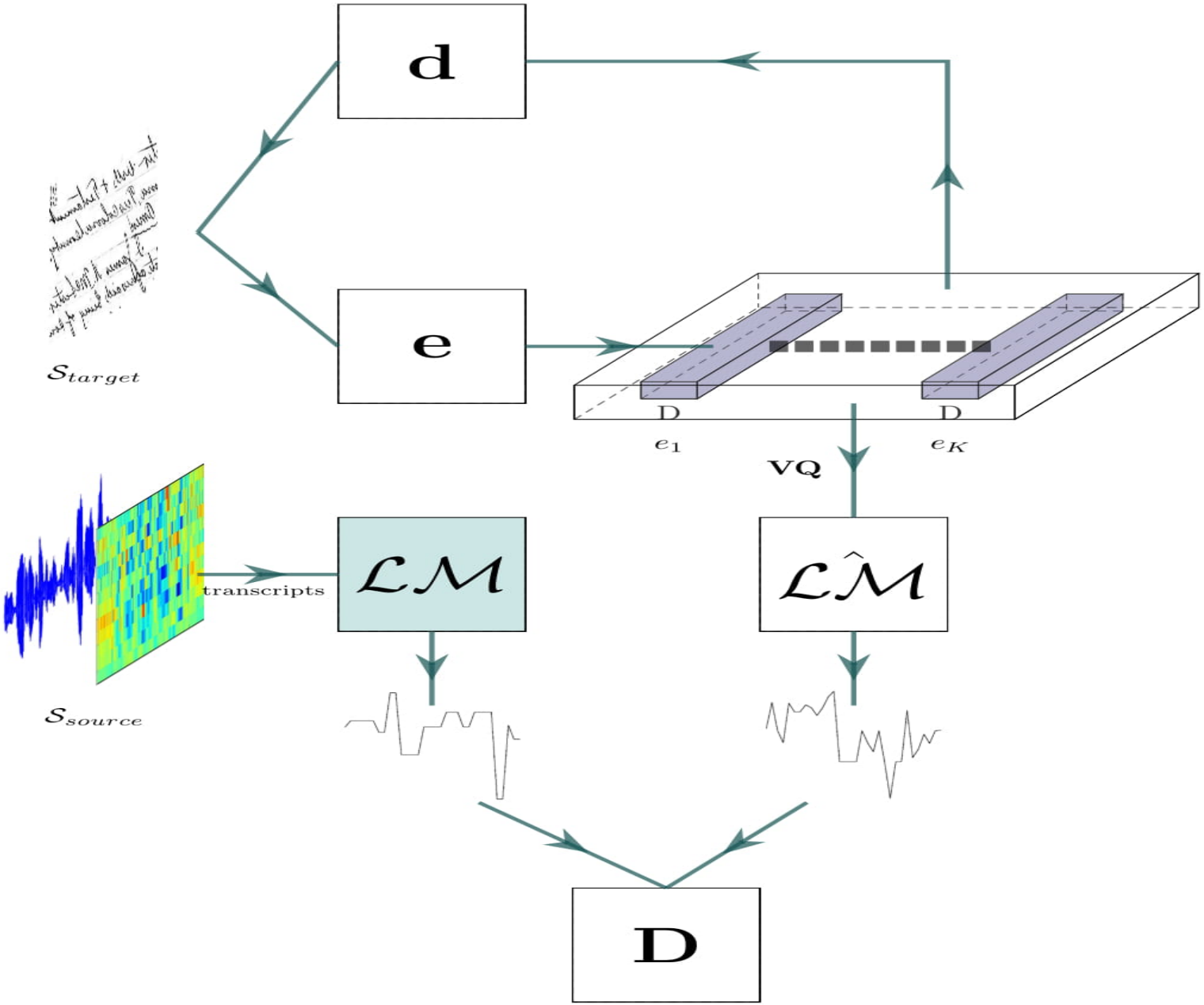


Figure: *Left* Audio frame points colored by speaker id and the embedding marks colored in black. *Right* Aaudio frame points colored by encoding indices and the embedding marks colored by number of embedding vectors.

Work achieved


d
e
$\mathcal{S}_{target}$
D
D
e_1
e_K
VQ
$\mathcal{S}_{source}$
transcripts
$\mathcal{LM}$
$\hat{\mathcal{LM}}$
D

Experiments

$$\mathbf{L}_a, \mathbf{M}_a \rightarrow \mathbf{L}_a, \mathbf{M}_a \qquad \mathbf{L}_a, \mathbf{M}_a \rightarrow \mathbf{L}_a, \mathbf{M}_b$$

Both train a Language Model $\mathcal{LM}$ computed from a dataset $\mathcal{S}_{\text{source}} \sim \mathbf{L}, \mathbf{M}$ and a Language Model $\mathcal{L}\hat{\mathcal{M}}$ computed from a dataset $\mathcal{S}_{\text{target}} \sim \mathbf{L}, \mathbf{M}$ and use them on the main model training with a discriminator $\mathbf{D}$.

n-gram Language Model

Compute the n-gram matrix $\mathcal{G}$ from the source dataset $\mathcal{S}_{\text{source}}$;

Compute the entropy vector $\mathcal{G}_H$ using $\mathcal{G}$;

Compute the entropy dataset $\mathcal{S}_{\mathcal{G}_H}$ using the entropy vector $\mathcal{G}_H$ using $\mathcal{G}$ and the source dataset $\mathcal{S}_{\text{source}}$;

Do the same steps for $\hat{\mathcal{G}}$ and $\mathcal{S}_{\hat{\mathcal{G}}_H}$ on $\mathcal{S}_{\text{target}}$.

n-gram Language Model

Compute the n-gram matrix $\mathcal{G}$ from the source dataset $\mathcal{S}_{\text{source}}$;

Compute the entropy vector $\mathcal{G}_H$ using $\mathcal{G}$;

Compute the entropy dataset $\mathcal{S}_{\mathcal{G}_H}$ using the entropy vector $\mathcal{G}_H$ using $\mathcal{G}$ and the source dataset $\mathcal{S}_{\text{source}}$;

Do the same steps for $\hat{\mathcal{G}}$ and $\mathcal{S}_{\hat{\mathcal{G}}_H}$ on $\mathcal{S}_{\text{target}}$.

n-gram Language Model

Compute the n-gram matrix $\mathcal{G}$ from the source dataset $\mathcal{S}_{\text{source}}$;

Compute the entropy vector $\mathcal{G}_H$ using $\mathcal{G}$;

Compute the entropy dataset $\mathcal{S}_{\mathcal{G}_H}$ using the entropy vector $\mathcal{G}_H$ using $\mathcal{G}$ and the source dataset $\mathcal{S}_{\text{source}}$;

Do the same steps for $\hat{\mathcal{G}}$ and $\mathcal{S}_{\hat{\mathcal{G}}_H}$ on $\mathcal{S}_{\text{target}}$.

n-gram Language Model

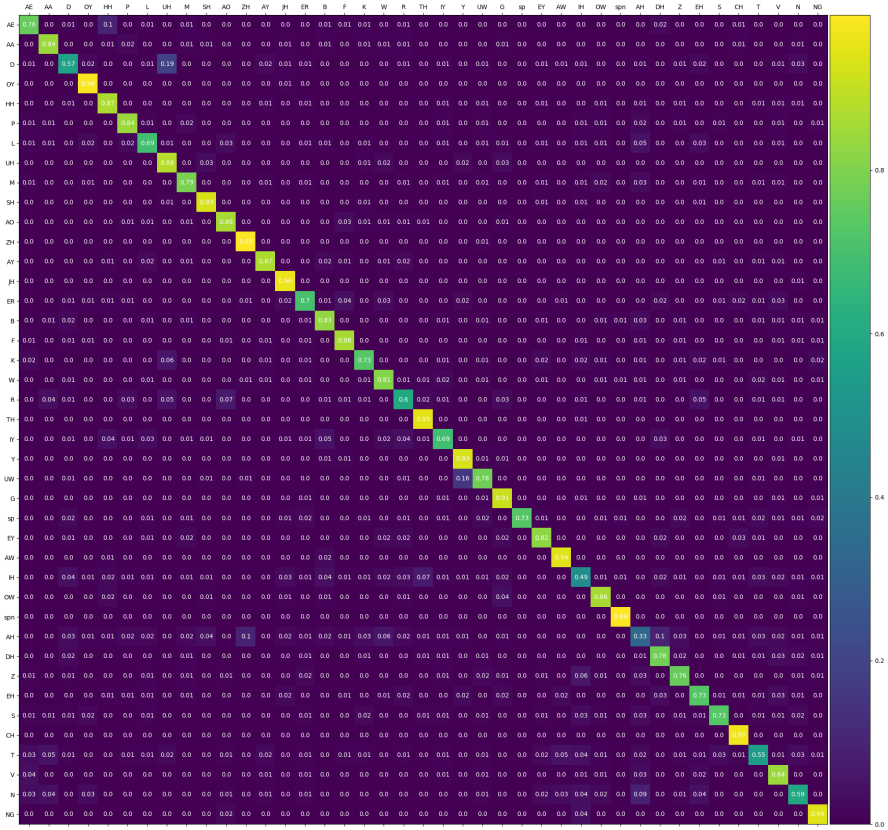
Compute the n-gram matrix $\mathcal{G}$ from the source dataset $\mathcal{S}_{\text{source}}$;

Compute the entropy vector $\mathcal{G}_H$ using $\mathcal{G}$;

Compute the entropy dataset $\mathcal{S}_{\mathcal{G}_H}$ using the entropy vector $\mathcal{G}_H$ using $\mathcal{G}$ and the source dataset $\mathcal{S}_{\text{source}}$;

Do the same steps for $\hat{\mathcal{G}}$ and $\mathcal{S}_{\hat{\mathcal{G}}_H}$ on $\mathcal{S}_{\text{target}}$.

Example of bi-gram Language Model



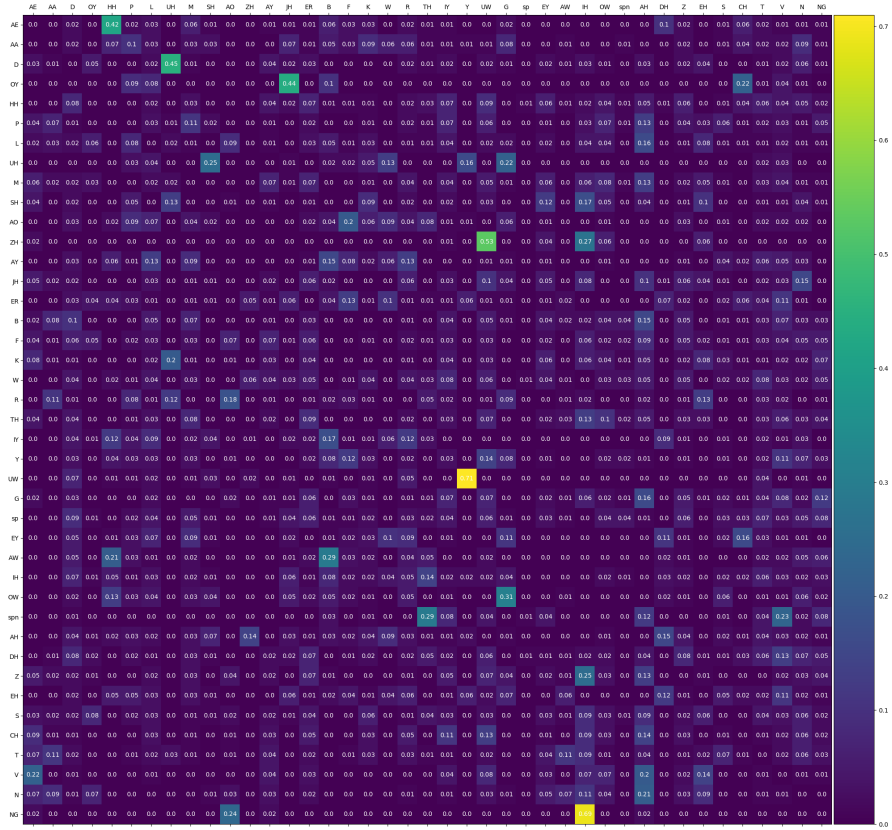


Figure: The bi-gram matrix of the groundtruth alignments of the validation subset of VCTK dataset, with a step of 20ms.

RNN Language Model

Train iteratively the model $\mathcal{R}$ on $\mathcal{S}_{\text{source}}$;

Compute the source entropy dataset $\mathcal{S}_{\mathcal{R}_H}$ from $\mathcal{R}$;

Do the same steps for $\hat{\mathcal{R}}$ and $\mathcal{S}_{\hat{\mathcal{R}}_H}$ on $\mathcal{S}_{\text{target}}$.

RNN Language Model

Train iteratively the model $\mathcal{R}$ on $\mathcal{S}_{\text{source}}$;

Compute the source entropy dataset $\mathcal{S}_{\mathcal{R}_H}$ from $\mathcal{R}$;

Do the same steps for $\hat{\mathcal{R}}$ and $\mathcal{S}_{\hat{\mathcal{R}}_H}$ on $\mathcal{S}_{\text{target}}$.

RNN Language Model

Train iteratively the model $\mathcal{R}$ on $\mathcal{S}_{\text{source}}$;

Compute the source entropy dataset $\mathcal{S}_{\mathcal{R}_H}$ from $\mathcal{R}$;

Do the same steps for $\hat{\mathcal{R}}$ and $\mathcal{S}_{\hat{\mathcal{R}}_H}$ on $\mathcal{S}_{\text{target}}$.

Adversarial training

Fooling loss: $\mathbf{BCE}(\mathbf{D}(\hat{y}^{(t)}_{\mathcal{G}_H}), 1)$;

Autoencoder training objective using ngram LM:

$$\min_{\mathbf{e},\mathbf{d}} \mathcal{L}_{\text{ngram}} = \mathcal{L}_{\text{VQ-VAE}} + \mathcal{L}_{\text{fool}}$$

Autoencoder training objective using RNN LM:

$$\min_{\mathbf{e},\mathbf{d}} \mathcal{L}_{\text{rnn}} = \mathcal{L}_{\text{VQ-VAE}} + \mathcal{L}_{\text{fool}} + \mathcal{L}_{\hat{\mathcal{R}}}$$

Discriminator training objective for both case:

$$\max_{\mathbf{D}} \mathcal{L}_{\mathbf{D}} = 0.5 \times (\mathbf{BCE}(\mathbf{D}(y^{(t)}_{\mathcal{G}_H} \in \mathcal{S}_{\mathcal{G}_H}), 1) + \mathbf{BCE}(\mathbf{D}(sg(\hat{y}^{(t)}_{\mathcal{G}_H})), 0))$$

Adversarial training

Fooling loss: $\mathbf{BCE}(\mathbf{D}(\hat{y}^{(t)}_{\mathcal{G}_H}), 1)$;

Autoencoder training objective using ngram LM:
$\min_{\mathbf{e},\mathbf{d}} \mathcal{L}_{\text{ngram}} = \mathcal{L}_{\text{VQ-VAE}} + \mathcal{L}_{\text{fool}}$

Autoencoder training objective using RNN LM:
$\min_{\mathbf{e},\mathbf{d}} \mathcal{L}_{\text{rnn}} = \mathcal{L}_{\text{VQ-VAE}} + \mathcal{L}_{\text{fool}} + \mathcal{L}_{\hat{\mathcal{R}}}$

Discriminator training objective for both case:
$\max_{\mathbf{D}} \mathcal{L}_{\mathbf{D}} = 0.5 \times (\mathbf{BCE}(\mathbf{D}(y^{(t)}_{\mathcal{G}_H} \in \mathcal{S}_{\mathcal{G}_H}), 1) + \mathbf{BCE}(\mathbf{D}(sg(\hat{y}^{(t)}_{\mathcal{G}_H})), 0))$

Adversarial training

Fooling loss: $\mathbf{BCE}(\mathbf{D}(\hat{y}^{(t)}_{\mathcal{G}_H}), 1)$;

Autoencoder training objective using ngram LM:
$\min_{\mathbf{e},\mathbf{d}} \mathcal{L}_{\text{ngram}} = \mathcal{L}_{\text{VQ-VAE}} + \mathcal{L}_{\text{fool}}$

Autoencoder training objective using RNN LM:
$\min_{\mathbf{e},\mathbf{d}} \mathcal{L}_{\text{rnn}} = \mathcal{L}_{\text{VQ-VAE}} + \mathcal{L}_{\text{fool}} + \mathcal{L}_{\hat{\mathcal{R}}}$

Discriminator training objective for both case:
$\max_{\mathbf{D}} \mathcal{L}_{\mathbf{D}} = 0.5 \times (\mathbf{BCE}(\mathbf{D}(y^{(t)}_{\mathcal{G}_H} \in \mathcal{S}_{\mathcal{G}_H}), 1) + \mathbf{BCE}(\mathbf{D}(sg(\hat{y}^{(t)}_{\mathcal{G}_H})), 0))$

Adversarial training

Fooling loss: $\mathbf{BCE}(\mathbf{D}(\hat{y}^{(t)}_{\mathcal{G}_H}), 1)$;

Autoencoder training objective using ngram LM:

$$\min_{\mathbf{e},\mathbf{d}} \mathcal{L}_{\text{ngram}} = \mathcal{L}_{\text{VQ-VAE}} + \mathcal{L}_{\text{fool}}$$

Autoencoder training objective using RNN LM:

$$\min_{\mathbf{e},\mathbf{d}} \mathcal{L}_{\text{rnn}} = \mathcal{L}_{\text{VQ-VAE}} + \mathcal{L}_{\text{fool}} + \mathcal{L}_{\hat{\mathcal{R}}}$$

Discriminator training objective for both case:

$$\max_{\mathbf{D}} \mathcal{L}_{\mathbf{D}} = 0.5 \times (\mathbf{BCE}(\mathbf{D}(y^{(t)}_{\mathcal{G}_H} \in \mathcal{S}_{\mathcal{G}_H}), 1) + \mathbf{BCE}(\mathbf{D}(sg(\hat{y}^{(t)}_{\mathcal{G}_H})), 0))$$

Evaluation and results

Evaluation

Let $u_1 = \{1, 1, 1, 2, 2, 3, 3\}$ and $v_1 = \{4, 4, 4, 1, 1, 3, 3\}$ be two samples of respectively subsets U and V;

$s = ARI(.,.)$ a clustering metric;

$ARI(u_1, v_1) = ARI(v_1, u_1) = 1.0.$

Evaluation

Let $u_1 = \{1, 1, 1, 2, 2, 3, 3\}$ and $v_1 = \{4, 4, 4, 1, 1, 3, 3\}$ be two samples of respectively subsets U and V;

$s = ARI(.,.)$ a clustering metric;

$$ARI(u_1, v_1) = ARI(v_1, u_1) = 1.0.$$

Evaluation

Let $u_1 = \{1, 1, 1, 2, 2, 3, 3\}$ and $v_1 = \{4, 4, 4, 1, 1, 3, 3\}$ be two samples of respectively subsets U and V;

$s = ARI(.,.)$ a clustering metric;

$ARI(u_1, v_1) = ARI(v_1, u_1) = 1.0.$

Results table

id	Architecture	ARI	AMI	Bottleneck accuracy	Perplexity
(i)	*basic*, random	0.0062	0.0028	n/a	12.8577
(ii)	*basic* + $J_p = 0.12$	0.0433	0.0203	n/a	37.9220
(iii)	*basic* + $\mathcal{S}_{\mathcal{G}_H^2}$ + $J_p = 0.12$, aligned, $\mathbf{M}_a$	0.0288	0.0244	n/a	n/a
(iv)	*pixcnn* + $J_p = 0.12$	0.1588	0.2785	0.6066	185.9
(v)	*pixcnn* + $\mathcal{S}_{\mathcal{G}_H^2}$, $J_p = 0.12$, aligned, $\mathbf{M}_a$	0.1825	0.2742	0.5767	63.7217
(vi)	*pixcnn* + $\mathcal{S}_{\mathcal{G}_H^2}$, $J_p = 0.12$, unaligned, $\mathbf{M}_a$	0.1784	0.2643	0.5670	59.53
(vii)	*pixcnn* + $\mathcal{S}_{\mathcal{G}_H^2}$, $J_p = 0.12$, unaligned, $\mathbf{M}_b$	0.1706	0.2704	0.5709	65.74
(viii)	*pixcnn* + $\mathcal{S}_{\mathcal{G}_H^3}$, $J_p = 0.12$, aligned, $\mathbf{M}_a$	0.1785	**0.2805**	0.5897	71.7067
(ix)	*pixcnn* + $\mathcal{S}_{\mathcal{G}_H^3}$, $J_p = 0.12$, unaligned, $\mathbf{M}_a$	0.1585	0.2645	0.5704	64.9483
(x)	*pixcnn* + $\mathcal{S}_{\mathcal{G}_H^3}$, $J_p = 0.12$, unaligned, $\mathbf{M}_b$	0.1836	0.2682	0.5777	67.6567
(xi)	*pixcnn* + $\mathcal{S}_{\mathcal{R}_H^4}$, $J_p = 0.25$, unaligned, $\mathbf{M}_b$	**0.1893**	0.2583	0.4940	51.5633

TABLE 1 – This figures shows the results obtained by the experiments described in the present section. J_p stands for the probability of jittering, if a jitter layer is used (cf. Section 3.3). $\mathcal{S}^4_{\mathcal{R}_H}$ stands for an entropy dataset computed from a RNN (cf. Section 3.6). $\mathbf{M}_a$ and $\mathbf{M}_b$ means the source and the target datasets have two distinct modalities. **ARI**, **AMI**, are respectively the Adjusted Rand Index and the Adjusted Mutual Information scores (cf. Section 3.7 and Appendix F). The language $\mathbf{L}$ of the datasets are always the same. Here *basic* uses 44 tokens and *pixcnn* 128 tokens in the bottleneck.

Results summary

Using VCTK (speech) as the source dataset and Scribblelens (handwritten) as the target dataset:

- Best ARI score using $\mathcal{S}_{\mathcal{R}_H^4}$;
- Best AMI score using $\mathcal{S}_{\mathcal{G}_H^3}$;
- These scores are slightly better than the state-of-the-art baseline.

Conclusion and perspectives

The results found on the ARI metric are consistent with our hypotheses (1) and (3);

The RNN has not reached its full capacity yet.

The results found on the ARI metric are consistent with our hypotheses (1) and (3);

The RNN has not reached its full capacity yet.

Hyperparameters tuning;

Repeat the previous experiments on multiple datasets;

Increase the size of k-seed;

Hyperparameters tuning;

Repeat the previous experiments on multiple datasets;

Increase the size of k-seed;

Hyperparameters tuning;

Repeat the previous experiments on multiple datasets;

Increase the size of k-seed;

Thank you for listening!

Example of obtained entropy curves