Unsupervised speech representation learning using WaveNet autoencoders

https://arxiv.org/abs/1901.08810

Jan Chorowski
University of Wrocław
06.06.2019
Deep Model = Hierarchy of Concepts

M. Zieler, “Visualizing and Understanding Convolutitional Networks”
Deep Learning history: 2006

2006: Stacked RBMs

Hinton, Salakhutdinov, “Reducing the Dimensionality of Data with Neural Networks”
Deep Learning history: 2012

2012: Alexnet
SOTA on Imagenet
Fully supervised training
Deep Learning Recipe

1. Get a massive, labeled dataset $D = \{(x, y)\}$:
   - Comp. vision: Imagenet, 1M images
   - Machine translation: EuroParlamanet data, CommonCrawl, several million sent. pairs
   - Speech recognition: 1000h (LibriSpeech), 12000h (Google Voice Search)
   - Question answering: SQuAD, 150k questions with human answers
   - ...

2. Train model to maximize $\log p(y|x)$
Value of Labeled Data

• Labeled data is crucial for deep learning

• But labels carry little information:
  – Example:
    An ImageNet model has 30M weights, but ImageNet is about 1M images from 1000 classes
    Labels: 1M * 10bit = 10Mbits

    Raw data: (128 x 128 images): ca 500 Gbits!
Value of Unlabeled Data

“The brain has about $10^{14}$ synapses and we only live for about $10^9$ seconds. So we have a lot more parameters than data. This motivates the idea that we must do a lot of unsupervised learning since the perceptual input (including proprioception) is the only place we can get $10^5$ dimensions of constraint per second.”

Geoff Hinton

https://www.reddit.com/r/MachineLearning/comments/2lmo0l/ama_geoffrey_hinton/
Unsupervised learning recipe

1. Get a massive labeled dataset \( D = \{ x \} \)
   Easy, unlabeled data is nearly free

2. Train model to...???
   What is the task?
   What is the loss function?
Unsupervised learning by modeling data distribution

Train the model to minimize $-\log p(x)$

E.g. in 2D:
- Let $D = \{x: x \in \mathbb{R}^2\}$
- Each point is an 2-dimensional vector
- We can draw a point-cloud
- And fit some known distribution, e.g. a Gaussian
Learning high dimensional distributions is hard

- Assume we work with small (32x32) images
- Each data point is a real vector of size $32 \times 32 \times 3$
- Data occupies only a tiny fraction of $\mathbb{R}^{32 \times 32 \times 3}$
- Difficult to learn!
Autoregressive Models

Decompose probability of data points in $R^n$ into $n$ conditional univariate probabilities:

$$p(x) = p(x_1, x_2, ..., x_n) = p(x_1)p(x_2|x_1) ... p(x_n|x_1, x_2, ..., x_{n-1}) = \prod_{i} p(x_i|x_{<i})$$
Autoregressive Example: Language modeling

Let $x$ be a sequence of word ids.

$$p(x) = p(x_1, x_2, ..., x_n) = \prod_{i} p(x_i | x_{<i})$$

$$\approx \prod_{i} p(x_i | x_{i-k}, x_{i-k+1}, ..., x_{i-1})$$

$p(\text{It’s a nice day}) = p(\text{It}) \times p(‘s | \text{it}) \times p(\text{a} | ‘s)\ldots$

- Classical n-gram models: cond. probs. estimated using counting
- Neural models: cond. probs. estimated using neural nets
WaveNet: Autoregressive modeling of speech

Treat speech as a sequence of samples!
Predict each sample based on previous ones.

https://arxiv.org/abs/1609.03499
PixelRNN:
A “language model for images”

Pixels generated left-to-right, top-to-bottom.

Cond. probabilities estimated using recurrent or convolutional neural networks.

PixelCNN samples

Salimans et al, “A PixelCNN Implementation with Discretized Logistic Mixture Likelihood and Other Modifications”
Autoregressive Models Summary

The good:
- Simple to define (pick an ordering).
- Often yield SOTA log-likelihood.

The bad:
- Training and generation require $O(n)$ ops.
- No compact intermediate data representation – not obvious how to use for downstream tasks.
Latent Variable Models

Intuition: to generate something complicated, do:

1. Sample something simple $z \sim \mathcal{N}(0,1)$

2. Transform it $x = \frac{z}{10} + \frac{z}{\|z\|}$
Variational autoencoder:
A neural latent variable model

Assume a 2 stage data generation process:

\[ z \sim \mathcal{N}(0,1) \]  
\[ x \sim p(x|z) \]

prior \( p(z) \) assumed to be simple
complicated transformation
implemented with a neural network

How to train this model?

\[ \log p(x) = \log \sum_z p(x|z)p(z) \]

This is often intractable!
ELBO: A lower bound on $\log p(x)$

Let $q(z|x)$ be any distribution. We can show that

$$\log p(x) =$$

$$= KL(q(z|x) \parallel p(z|x)) + \mathbb{E}_{z \sim q(z|x)} \left[ \log \left( \frac{p(z|x)}{q(z|x)} p(x) \right) \right]$$

$$\geq \mathbb{E}_{z \sim q(z|x)} \left[ \log \left( \frac{p(z|x)}{q(z|x)} p(x) \right) \right]$$

$$= \mathbb{E}_{z \sim q(z|x)} [\log p(x|z)] - KL(q(z|x) \parallel p(z))$$

The bound is tight for $p(z|x) = q(z|x)$. 
ELBO interpretation

ELBO, or evidence lower bound:

$$\log p(x) \geq \mathbb{E}_{z \sim q(z|x)}[\log p(x|z)] - KL(q(z|x) \parallel p(z))$$

where:

$$\mathbb{E}_{z \sim q(z|x)}[\log p(x|z)]$$ reconstruction quality:
how many nats we need to reconstruct $x$, when someone gives us $q(z|x)$

$$KL(q(z|x) \parallel p(z))$$ code transmission cost:
how many nats we transmit about $x$ in $q(z|x)$ rather than $p(z)$

Interpretation: do well at reconstructing $x$, limiting the amount of information about $x$ encoded in $z$. 

The Variational Autoencoder

\[ p(z) \xrightarrow{} KL(q(z|x) \parallel p(z)) \xleftarrow{} x \]

An input \( x \) is put through the \( q \) network to obtain a distribution over latent code \( z \), \( q(z|x) \).

Samples \( z_1, \ldots, z_k \) are drawn from \( q(z|x) \). They \( k \) reconstructions \( p(x|z_k) \) are computed using the network \( p \).
VAE is an Information Bottleneck

Each sample is represented as a Gaussian

This discards information (latent representation has low precision)
VQVAE – deterministic quantization

Limit precision of the encoding by quantizing (round each vector to a nearest prototype).

Output can be treated:
- As a sequence of discrete prototype ids (tokens)
- As a distributed representation (the prototypes themselves)

Train using the straight-through estimator, with auxiliary losses:

$$\mathcal{L} = \log p(x \mid z_q(x)) + \|\text{sg}(z_e(x)) - e_q(x)\|_2^2 + \gamma \|z_e(x) - \text{sg}(e_q(x))\|_2^2$$
VAEs and sequential data

To encode a long sequence, we apply the VAE to chunks:

But neighboring chunks are similar!
We are encoding the same information in many zs!
We are wasting capacity!
WaveNet + VAE

A WaveNet reconstructs the waveform using the information from the past.

Latent representations are extracted at regular intervals.

The WaveNet uses information from:

1. The past recording
2. The latent vectors $z$
3. Other conditioning, e.g. about speaker

The encoder transmits in $zs$ only the information that is missing from the past recording.

The whole system is a very low bitrate codec (roughly 0.7kbits/sec, the waveform is 16k Hz* 8bit=128kbit/sec)

van den Oord et al. Neural discrete representation learning
VAE + autoregressive models: latent collapse danger

- Purely Autoregressive models: SOTA log-likelihoods
- Conditioning on latents: information passed through bottleneck lower reconstruction x-entropy
- In standard VAE model actively tries to
  - reduce information in the latents
  - maximally use autoregressive information
  => Collapse: latents are not used!
- Solution: stop optimizing KL term (free bits), make it a hyperparam (VQVAE)
Model description

WaveNet decoder conditioned on:
- latents extracted at 24Hz-50Hz
- speaker

3 bottleneck evaluated:
- Dimensionality reduction, max 32 bits/dim
- VAE, $KL(q(z|x) \parallel \mathcal{N}(0,1))$ nats (bits)
- VQVAE with $K$ protos: $\log_2 K$ bits

Input:
Waveforms, Mel Filterbanks, MFCCs

Hope: speaker separated form content.
We have inserted probing classifiers at 4 points in the network:

\[ p_{\text{cond}} \]: several \( z \) codes mixed together using a convolution. The wavenet uses it for conditioning

\[ p_{\text{bn}} \]: the latent codes

\[ p_{\text{proj}} \]: low dimensional representation input to the bottleneck layer

\[ p_{\text{enc}} \]: high dimensional representation coming out of the encoder
Experimental Questions

• What information is captured in the latent codes/probing points?
• What is the role of the bottleneck layer?
• Can we regularize the latent representation?
• How to promote a segmentation?
• How good is the representation on downstream tasks?
• What design choices affect it?
VQVAE Latent representation
What information is captured in the latent codes?

For each probing point, we have trained predictors for:

- Framewise phoneme prediction
- Speaker prediction
- Gender prediction
- Mel Filterbank reconstruction
Results

![Graph showing results for different datasets with varying latent dimensions.](image-url)
Phonemes vs Gender tradeoff

![Graph showing phoneme prediction accuracy vs gender prediction accuracy with various lines representing different models and probe points.](image)
How to regularize the latent codes?

We want the codes to capture phonetic information.

Phones vary in duration – from about 30ms to 1s (long silences).

Thus we need to extract the latent codes frequently enough to capture the short phones, but when the phone doesn’t change, the latents should be stable too.

This is similar to slow features analysis.
Problem with enforcing slowness

Enforcing slow features (small changes to the latents), has a trivial optimum: constant latents.

Then WaveNet can just disregard the encoder, and latent space collapses.
Randomized time jitter
Rather than putting a penalty on changes of the latent $z$ vectors, add time jitter to them. This forces the model to have a more stable representation over time.
Randomized time jitter results
How to learn a segmentation?

The representation should be constant within a phoneme, then change abruptly

Enforcing slowness leads to collapse, jitter prevents the model from using pairs of tokens as codepoints

Idea: allow the model to infrequently emit a non-trivial representation
Non-max suppression – choosing where to emit

Latents computed at 25Hz, but allow only \( \frac{1}{4} \) nonzero
Non-max suppression – choosing where to emit

Token 13 is near emissions of "S" and some "Z"
Non-max suppression – choosing where to emit

Token 17 is near emissions of some „L”
Performance on ZeroSpeech unit discovery

<table>
<thead>
<tr>
<th>Model</th>
<th>English (45h) 1s</th>
<th>English (45h) 10s</th>
<th>English (45h) 2m</th>
<th>French (24h) 1s</th>
<th>French (24h) 10s</th>
<th>French (24h) 2m</th>
<th>Mandarin (2.4h) 1s</th>
<th>Mandarin (2.4h) 10s</th>
<th>Mandarin (2.4h) 2m</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unsupervised baseline</td>
<td>12.0</td>
<td>12.1</td>
<td>12.1</td>
<td>12.5</td>
<td>12.6</td>
<td>12.6</td>
<td>11.5</td>
<td>11.5</td>
<td>11.5</td>
</tr>
<tr>
<td>Supervised topline</td>
<td>6.5</td>
<td>5.3</td>
<td>5.1</td>
<td>8.0</td>
<td>6.8</td>
<td>6.8</td>
<td>9.5</td>
<td>4.2</td>
<td>4.0</td>
</tr>
<tr>
<td>VQ-VAE (per lang, $p_{cond}$)</td>
<td><strong>5.6</strong></td>
<td><strong>5.5</strong></td>
<td><strong>5.5</strong></td>
<td><strong>7.3</strong></td>
<td><strong>7.5</strong></td>
<td><strong>7.5</strong></td>
<td><strong>11.2</strong></td>
<td><strong>10.7</strong></td>
<td><strong>10.8</strong></td>
</tr>
<tr>
<td>Heck et al. [57]</td>
<td>6.9</td>
<td>6.2</td>
<td>6.0</td>
<td>9.7</td>
<td>8.7</td>
<td>8.4</td>
<td><strong>8.8</strong></td>
<td><strong>7.9</strong></td>
<td><strong>7.8</strong></td>
</tr>
<tr>
<td>Chen et al. [58]</td>
<td>8.5</td>
<td>7.3</td>
<td>7.2</td>
<td>11.2</td>
<td>9.4</td>
<td>9.4</td>
<td>10.5</td>
<td>8.7</td>
<td>8.5</td>
</tr>
<tr>
<td>Ansari et al. [59]</td>
<td>7.7</td>
<td>6.8</td>
<td>N/A</td>
<td>10.4</td>
<td>N/A</td>
<td>8.8</td>
<td>10.4</td>
<td>9.3</td>
<td>9.1</td>
</tr>
<tr>
<td>Yuan et al. [60]</td>
<td>9.0</td>
<td>7.1</td>
<td>7.0</td>
<td>11.9</td>
<td>9.5</td>
<td>9.5</td>
<td>11.1</td>
<td>8.5</td>
<td>8.2</td>
</tr>
<tr>
<td>Across-speaker</td>
<td>English (45h)</td>
<td>French (24h)</td>
<td>Mandarin (2.4h)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1s 10s 2m</td>
<td>1s 10s 2m</td>
<td>1s 10s 2m</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unsupervised baseline</td>
<td>23.4</td>
<td>23.4</td>
<td>23.4</td>
<td>25.2</td>
<td>25.5</td>
<td>25.2</td>
<td>21.3</td>
<td>21.3</td>
<td>21.3</td>
</tr>
<tr>
<td>Supervised topline</td>
<td>8.6</td>
<td>6.9</td>
<td>6.7</td>
<td>10.6</td>
<td>9.1</td>
<td>8.9</td>
<td>12.0</td>
<td>5.7</td>
<td>5.1</td>
</tr>
<tr>
<td>VQ-VAE (per lang, $p_{cond}$)</td>
<td><strong>8.1</strong></td>
<td><strong>8.0</strong></td>
<td><strong>8.0</strong></td>
<td><strong>11.0</strong></td>
<td><strong>10.8</strong></td>
<td><strong>11.1</strong></td>
<td><strong>12.2</strong></td>
<td><strong>11.7</strong></td>
<td><strong>11.9</strong></td>
</tr>
<tr>
<td>Heck et al. [57]</td>
<td>10.1</td>
<td>8.7</td>
<td>8.5</td>
<td>13.6</td>
<td>11.7</td>
<td>11.3</td>
<td><strong>8.8</strong></td>
<td><strong>7.4</strong></td>
<td><strong>7.3</strong></td>
</tr>
<tr>
<td>Chen et al. [58]</td>
<td>12.7</td>
<td>11.0</td>
<td>10.8</td>
<td>17.0</td>
<td>14.5</td>
<td>14.1</td>
<td>11.9</td>
<td>10.3</td>
<td>10.1</td>
</tr>
<tr>
<td>Ansari et al. [59]</td>
<td>13.2</td>
<td>12.0</td>
<td>N/A</td>
<td>17.2</td>
<td>N/A</td>
<td>15.4</td>
<td>13.0</td>
<td>12.2</td>
<td>12.3</td>
</tr>
<tr>
<td>Yuan et al. [60]</td>
<td>14.0</td>
<td>11.9</td>
<td>11.7</td>
<td>18.6</td>
<td>15.5</td>
<td>14.9</td>
<td>12.7</td>
<td>10.8</td>
<td>10.7</td>
</tr>
</tbody>
</table>

SOTA results in unsupervised phoneme discrimination Fr and EN ZeroSpeech challenge.

Mandarin shows limitation of the method:
- Too little training data (only 2.4h unsup. speech)
- Tonal information is discarded.
The quantization discards speaker info, improving across-speaker results. MFCCs slightly better than FBanks.
Mandarin: VQVAE bottleneck discards phone information

<table>
<thead>
<tr>
<th>Mandarin</th>
<th>Within spkr.</th>
<th>Across spkr.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VQ-VAE (per lang, MFCC, $p_{cond}$)</td>
<td>11.2</td>
<td>12.2</td>
</tr>
<tr>
<td>VQ-VAE (per lang, MFCC, $p_{bn}$)</td>
<td>10.8</td>
<td>11.9</td>
</tr>
<tr>
<td>VQ-VAE (per lang, MFCC, $p_{proj}$)</td>
<td>9.9</td>
<td>11.0</td>
</tr>
<tr>
<td>VQ-VAE (all lang, MFCC, $p_{cond}$)</td>
<td>9.2</td>
<td>10.3</td>
</tr>
<tr>
<td>VQ-VAE (all lang, MFCC, $p_{bn}$)</td>
<td>9.0</td>
<td>9.9</td>
</tr>
<tr>
<td>VQ-VAE (all lang, MFCC, $p_{proj}$)</td>
<td>7.4</td>
<td>8.6</td>
</tr>
<tr>
<td>VQ-VAE (all lang, fbank, $p_{proj}$)</td>
<td>6.8</td>
<td>7.8</td>
</tr>
</tbody>
</table>

The quantization discards too much (tone insensitivity?)
MFCCs worse than FBanks
What impacts the representation?

Implicit time constant of the model:

• Input field of view of the encoder – optimum close to 0.3s

• WaveNet field of view - needs at minimum 10ms
Failed attempts

- I found no benefits from building a hierarchical representation (extract latents at different timescales), even when the slower latents had no bottleneck
- Filterbank reconstruction works worse than waveform
  - Too easy for the autoregressive model?
  - Too little detail?
The future

We will explore similar ideas during JSALT2019 topic “Distant supervision for representation learning”.

The workshop will:
- Work on speech and handwriting
- Explore ways of integrating metadata and unlabeled data to control latent representations
- Focus on downstream supervised OCR and ASR tasks under low data conditions

Some approaches to try:
- Contrastive predictive coding
- Masked reconstruction
The future: CPC

- Contrastive coding learns representations that can tell a frame from other ones.

Oord et al. „Representation Learning with Contrastive Predictive Coding”
Schneider et al. „wav2vec: Unsupervised Pre-training for Speech Recognition”
The future: masked reconstruction

• BERT is a recent, SOTA model for sentence representation learning

• Mask the inputs:

Labels: [MASK]₁ = store; [MASK]₂ = gallon
Thank you!

• Questions?
Backup
ELBO Derivation pt. 1

\[ KL(q_\phi(z|x) \| p_\theta(z|x)) = \mathbb{E}_{z \sim q_\phi(z|x)} \left[ -\log \frac{p_\theta(z|x)}{q_\phi(z|x)} \right] = \]

\[ = \mathbb{E}_{z \sim q_\phi(z|x)} \left[ -\log \frac{p_\theta(z|x)p_\theta(x)}{q_\phi(z|x)p_\theta(x)} \right] = \]

\[ = \mathbb{E}_{z \sim q_\phi(z|x)} \left[ -\log \frac{p_\theta(x,z)}{q_\phi(z|x)} \right] + \log p_\theta(x) \]

\[ \log p_\theta(x) = KL(q_\phi(z|x) \| p_\theta(z|x)) + \mathbb{E}_{z \sim q_\phi(z|x)} \left[ \log \frac{p_\theta(x,z)}{q_\phi(z|x)} \right] \]
ELBO derivation pt. 2

\[
\log p_\theta(x) \geq \mathbb{E}_{z \sim q_\phi(z|x)} \left[ \log \frac{p_\theta(x, z)}{q_\phi(z|x)} \right] = \\
= \mathbb{E}_{z \sim q_\phi(z|x)} \left[ \log \frac{p_\theta(x|z)p_\theta(z)}{q_\phi(z|x)} \right] = \\
= \mathbb{E}_{z \sim q_\phi(z|x)} [\log p_\theta(x|z)] - \mathbb{E}_{z \sim q_\phi(z|x)} \left[ - \log \frac{p_\theta(z)}{q_\phi(z|x)} \right] \\
= \mathbb{E}_{z \sim q_\phi(z|x)} [\log p_\theta(x|z)] - KL(q_\phi(z|x)\|p_\theta(z))
\]