M2 Data Science

DS-telecom-15 "Audio and Music Information Retrieval"

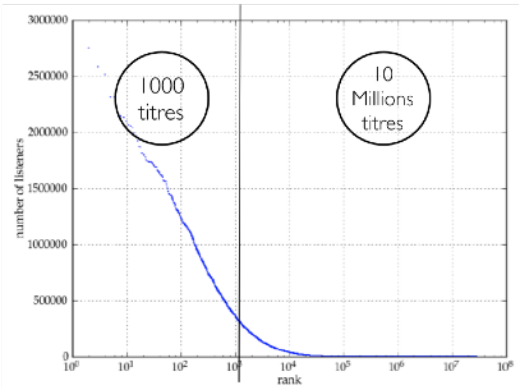
Geoffroy Peeters

contact: geoffroy.peeters@telecom-paris.fr
Télécom-Paris, IP-Paris, France

Music classification, segmentation and search-by-similarity

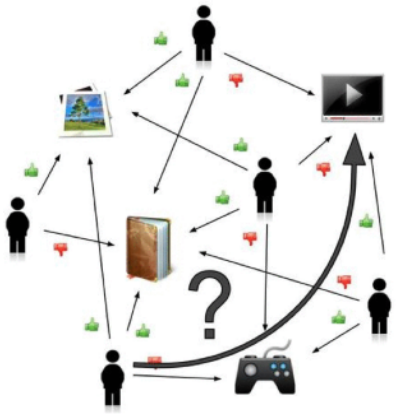
Why do we need auto-tagging ?

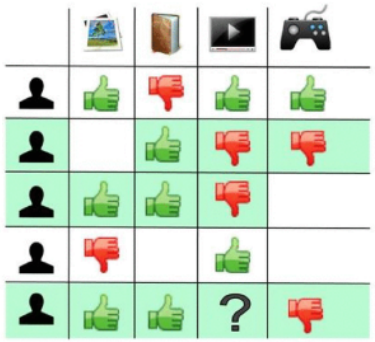
The cold start problem



- **The long-tail**
  - few music tracks are listened a lot
  - a lot of music tracks are listened few
  - → we will **recommend** these tracks to the user

- **Collaborative filtering recommendation**
  - if a person A has the same opinion as a person B on an issue, A is more likely to have B's opinion on a different issue than that of a randomly chosen person





- **Cold-start problem**
  - music tracks which nobody listen to do not have any **user** data → not possible to use collaborative-filtering recommendation
    - → content-based recommendation

      (1) tag-based

      (2) distance-based

source: https://en.wikipedia.org/wiki/Collaborative_filtering

Music track description

Editorial meta-data

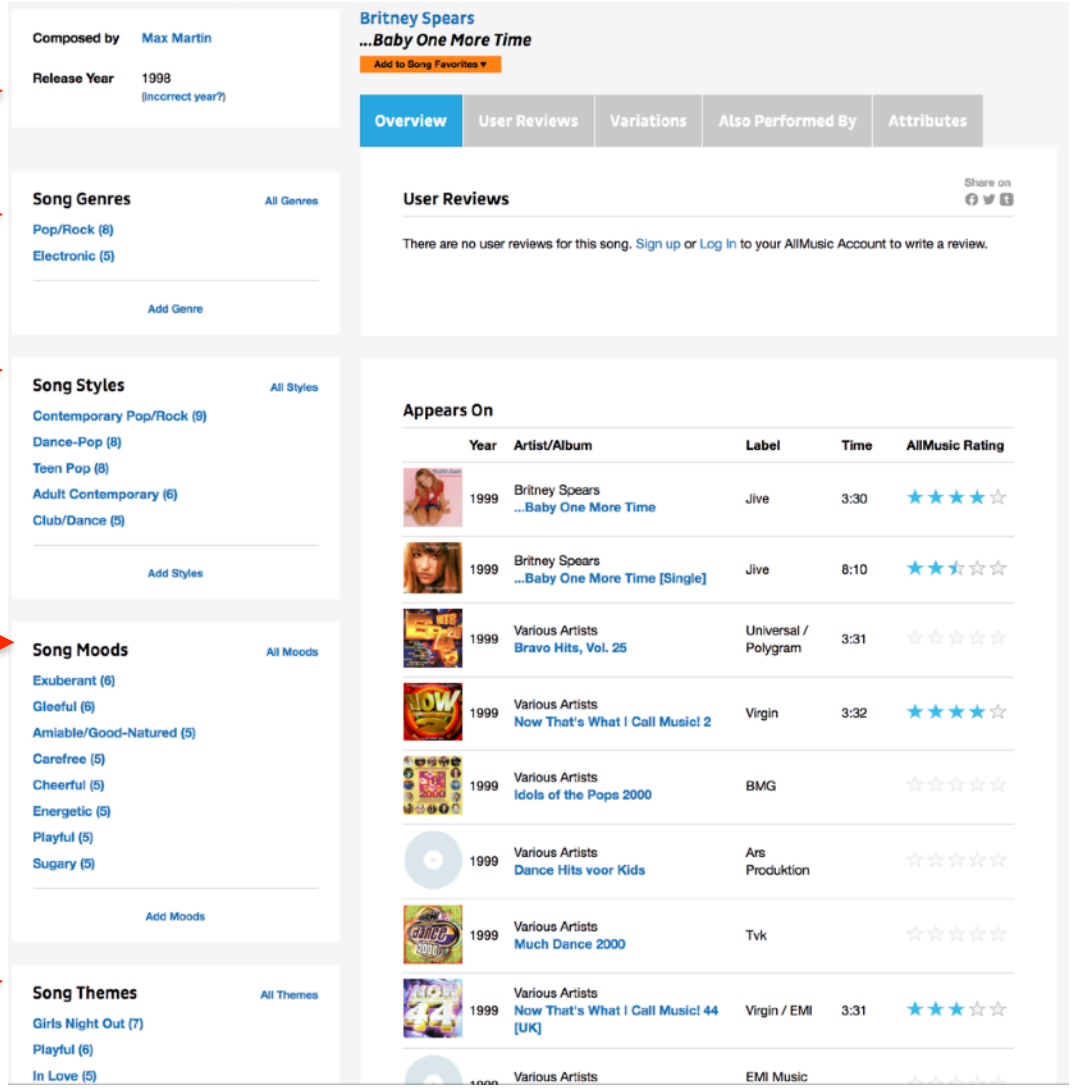
year →

genre →

styles →

moods →

theme →



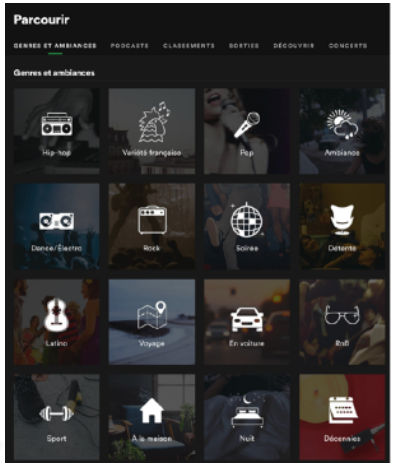
Music track description

Editorial meta-data: genre taxonomy

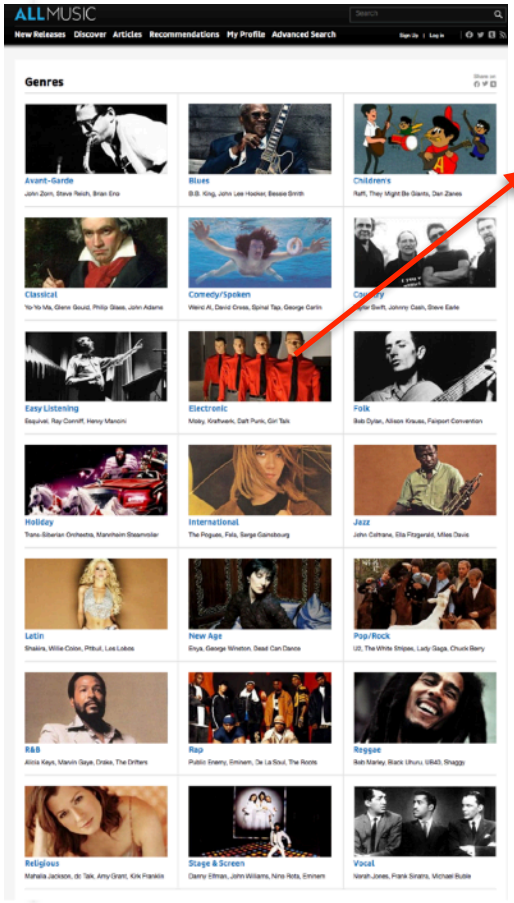
– not shared across labels, services

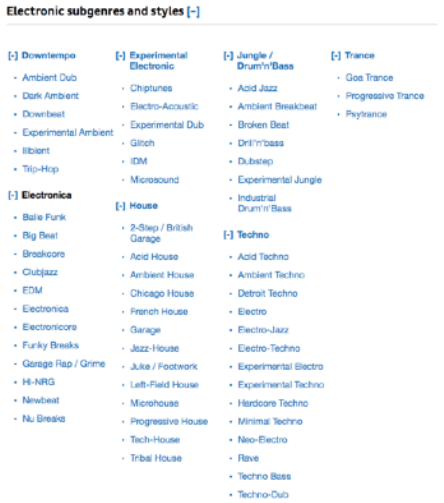
Deezer

Spotify

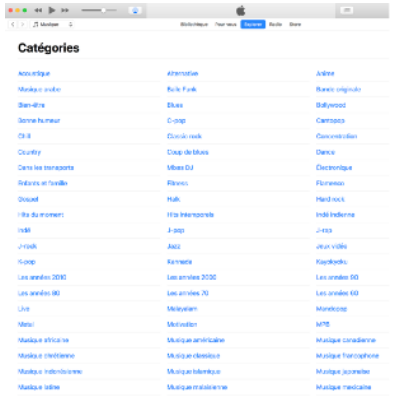


All Music Guide





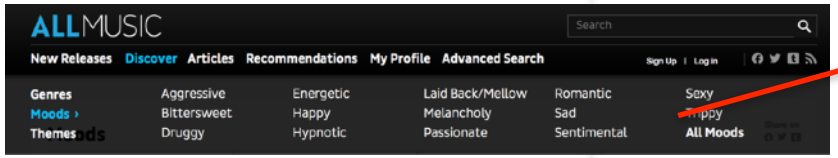
Apple music

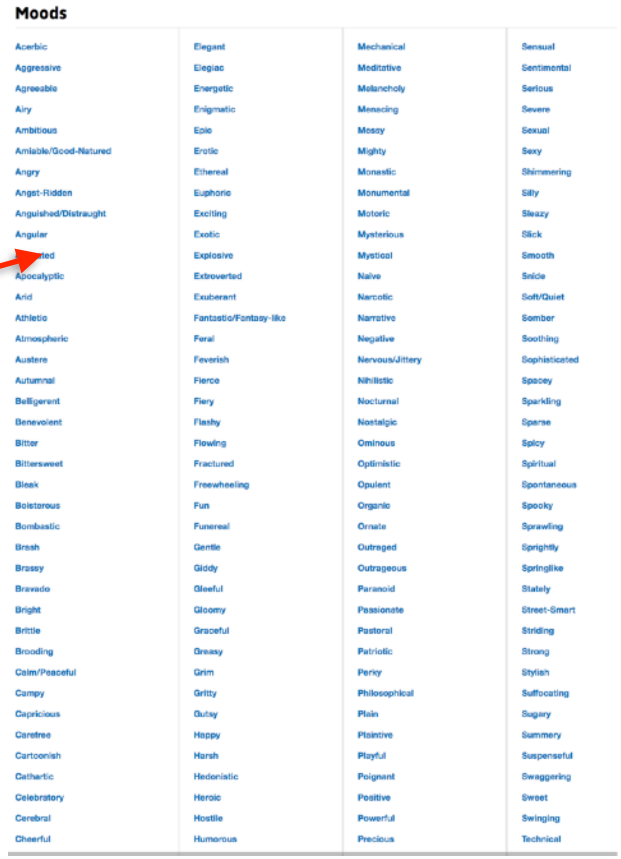


Music track description

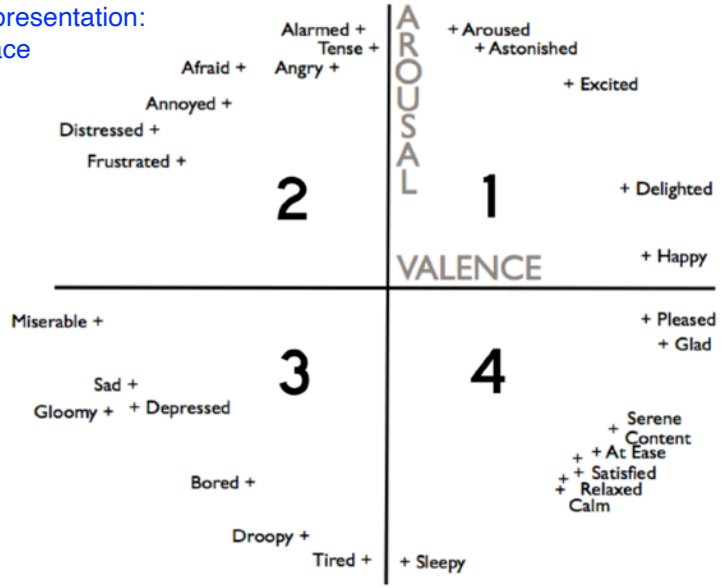
Editorial meta-data: mood taxonomy

– not shared across labels, services





Continuous value representation:
Valence/ Arousal space

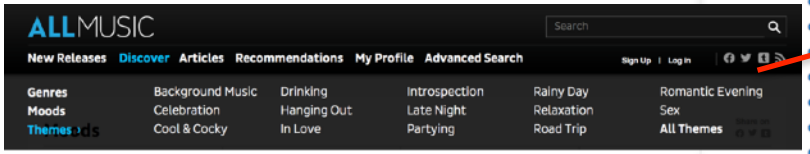


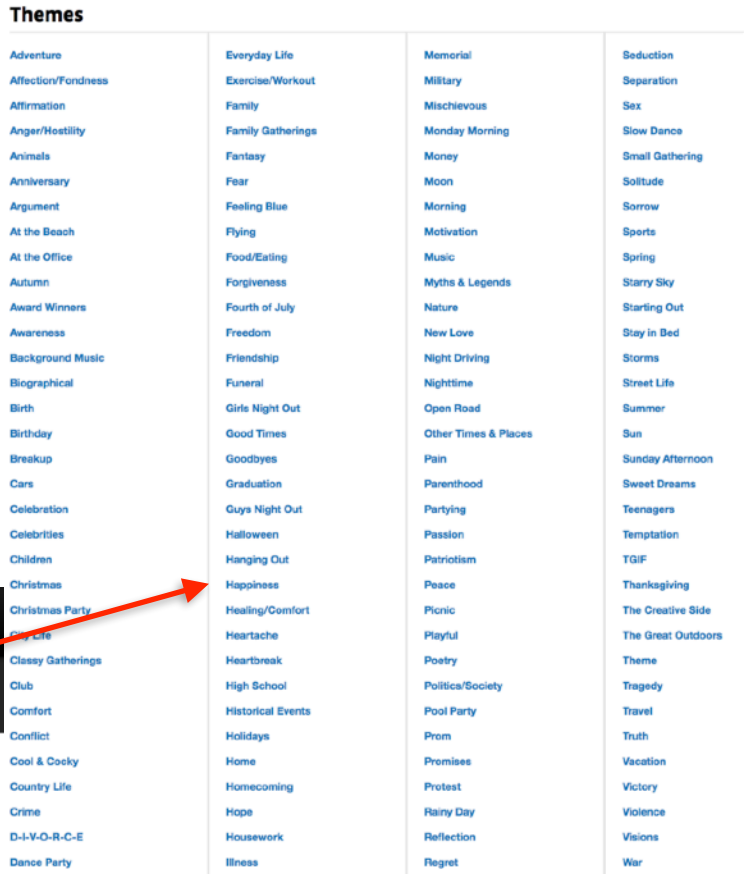
"Circumplex model of affect" with arousal and valence dimensions, adapted from Russell (1980)

Music track description

Editorial meta-data: context/usage/theme taxonomy

– not shared across labels, services

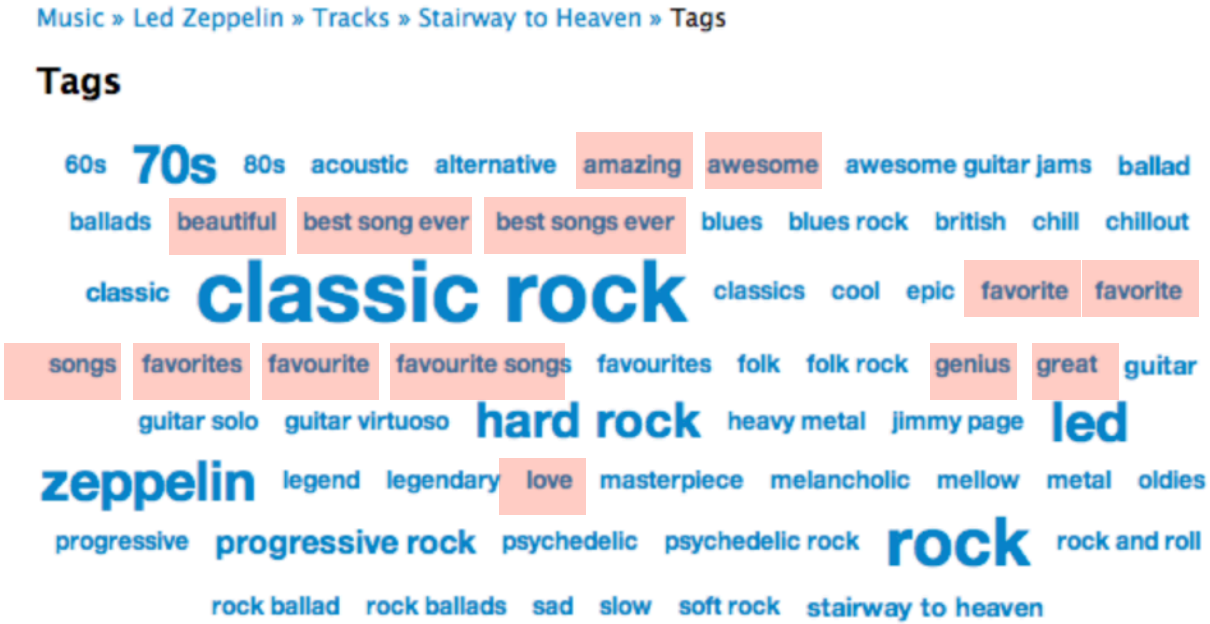




Music track description

User meta-data = tags = folksonomy ≠ taxonomy

– numerous, cost-free to get but very noisy !!!



Deep Learning reminders

Deep Learning reminders

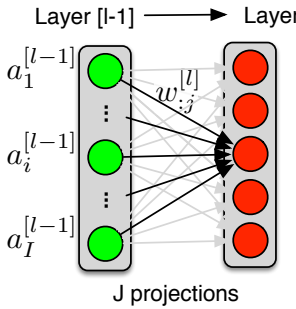
Architectures

Multi-Layer-Perceptron (MLP)

- Extension of the Perception

- Neurons are organized into Layers which are Fully-Connected
  - $a_j^{[l]}$ is connected to all neurons $a_i^{[l-1]}$
  - connection done through
    - multiplications by weights $w_{ij}^{[l]}$,
    - addition of a bias $b_j^{[l]}$,
    - passing through non-linear activation $g(.)$
  - Each $\mathbf{w}_j^{[l]}$ defines a specific projection of the previous layer

Fully Connected Neural Network



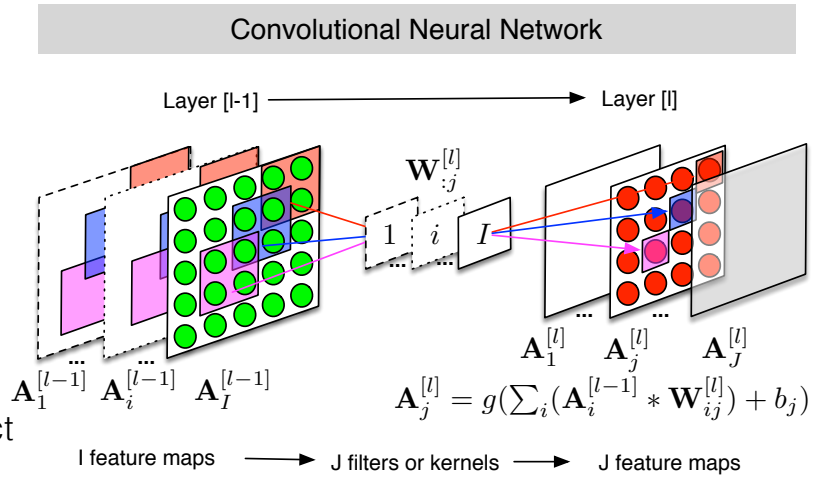
$$a_j^{[l]} = g(\sum_i a_i^{[l-1]} w_{ij}^{[l]} + b_j^{[l]})$$

$$\mathbf{a}^{[l]} = g(\mathbf{a}^{[l-1]} \mathbf{W}^{[l]} + \mathbf{b}^{[l]})$$

Frank Rosenblatt. The perceptron, a perceiving and recognizing automaton Project Para. Cornell Aeronautical Laboratory, 1957.

Convolutional Neural Network (CNN)

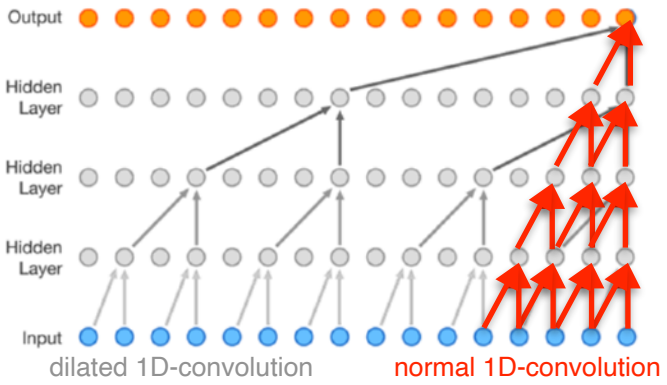
- (1) Assume local connectivity of neurons of a given layer (vision: nearby pixels more correlated)
  - Each region (x, y) of $\mathbf{A}_i^{[l-1]}$ is projected individually
  - Results= feature map noted $\mathbf{A}_{i \to j}^{[l]}$ for the j^{th} projection
- (2) Add a **parameter sharing** property
  - for a given j, the same projection $\mathbf{W}_{ij}^{[l]}$ is used to project the different regions $(x, y) \rightarrow$ the weights are shared
  - apply the same projection to the various regions (x, y)
- (1)+(2) $\rightarrow$ convolution operator
- In practice
  - several input feature maps $\mathbf{A}_{1 \ldots i \ldots I}^{[l-1]}$, projection with a tensor $\mathbf{W}_{:j}^{[l]}$ (extends over I)
  - Results: feature map $\mathbf{A}_j^{[l]}$
  - Several projections: J different convolutions resulting in J output feature maps
- To reduce dimensionality, allows spatial invariance:
  - **max-pooling**



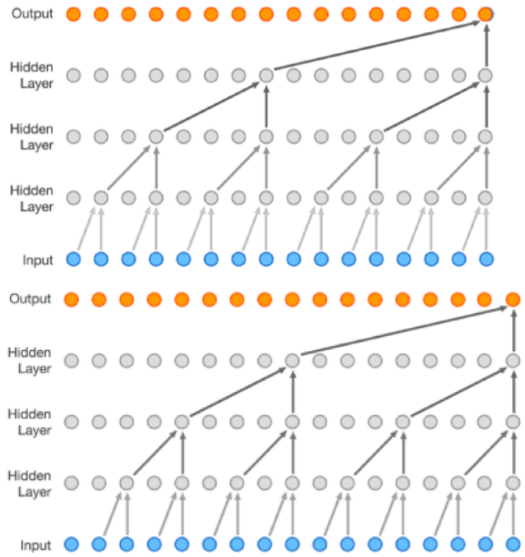
- Kunihiko Fukushima and Sei Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In Competition and cooperation in neural nets, pages 267–285. Springer, 1982.
- Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haner. Gradient- based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324, 1998

Temporal Convolutional Networks (TCN)

- Motivation: learn better projections than Fourier
- 1D-convolution applied on the raw audio waveform $x(n)$
  - filters $\mathbf{W}_j$ have only one dimension (time)
  - convolution is done over time



- Receptive field (RC)
  - portion of the input data to which a given neuron responds
  - images (256x256): only a few layers is necessary in CV to make the RC of a neuron cover the whole input image
  - audio (44100/sec): requires a huge number of layers



- 1D-Dilated- Convolutions
  - $(x \circledast_d w)(n) = \sum_{i=0}^{l-1} w(i)x(n-(d\,.\,i))$
  - the filters is convolved with the signal only considering one over d values

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. arXiv preprint arXiv:1609.03499, 2016.

Temporal Convolutional Networks (TCN)

- 1D-Convolution
- + causality constraint
- + stacks two dilated-convolutions on top of each other (with weight-normalization, ReLu, DropOut)
- + with a parallel residual path

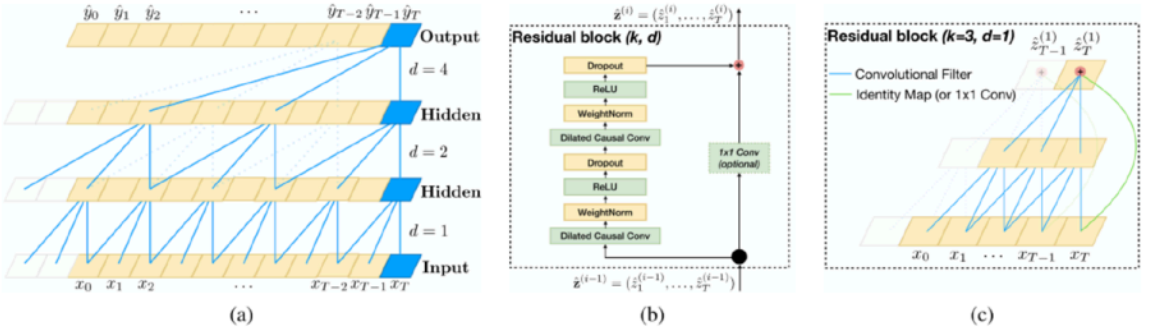


Figure 1. Architectural elements in a TCN. (a) A dilated causal convolution with dilation factors $d = 1, 2, 4$ and filter size $k = 3$. The receptive field is able to cover all values from the input sequence. (b) TCN residual block. An 1x1 convolution is added when residual input and output have different dimensions. (c) An example of residual connection in a TCN. The blue lines are filters in the residual function, and the green lines are identity mappings.

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv preprint arXiv:1803.01271, 2018.

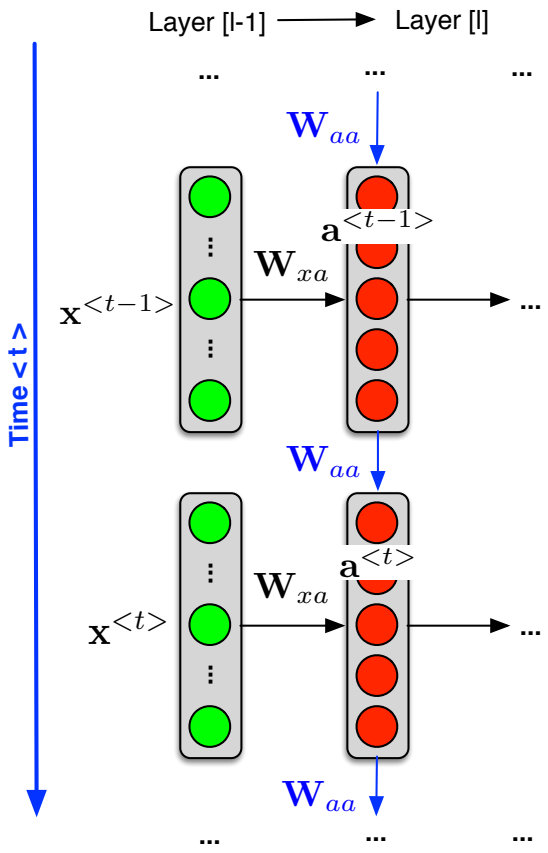
Recurrent Neural Network (RNN)

- RNNs take into account the sequential aspect of the data
  - RNNs have a memory that keeps track of previously proposed events of the sequence
  - Internal/hidden representation of the data at time t, $\mathbf{a}^{<t>}$ does not only depend on the input data $\mathbf{x}^{<t>}$ but also on the internal/hidden representation at the previous time $\mathbf{a}^{<t-1>}$

- More sophisticated RNN cells
  - Long Short Term Memory (LSTM)
  - Gated Recurrent Units (GRU)

Recurrent Neural Network

$$\mathbf{a}^{<t>} = g(\mathbf{x}^{<t>}\mathbf{W}_{xa} + \mathbf{a}^{<t-1>}\mathbf{W}_{aa} + \mathbf{b}_a)$$



- David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. nature, 323(6088):533–536, 1986.
- Sepp Hochreiter and Jurgen Schmidhuber. Long short-term memory. Neural computation, 9(8):1735–1780, 1997.
- Kyunghyun Cho, Bart Van Merri¨enboer, Caglar Gulcehre, Dzmitry Bah- danau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078, 2014.

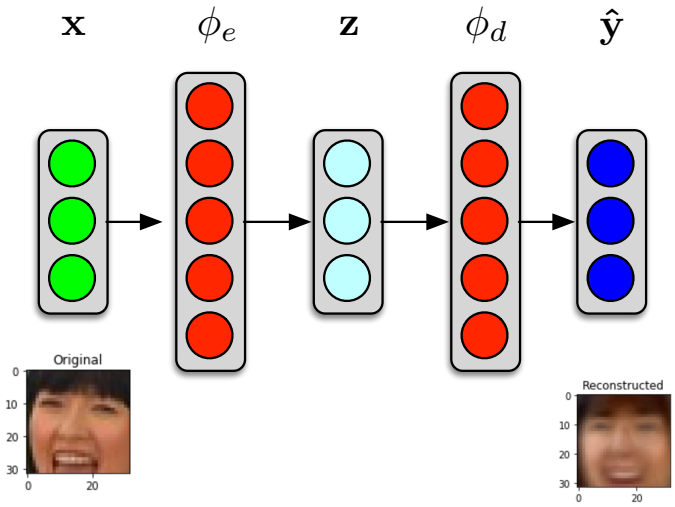
Deep Learning reminders

Meta-architectures

Auto-Encoder (AE)

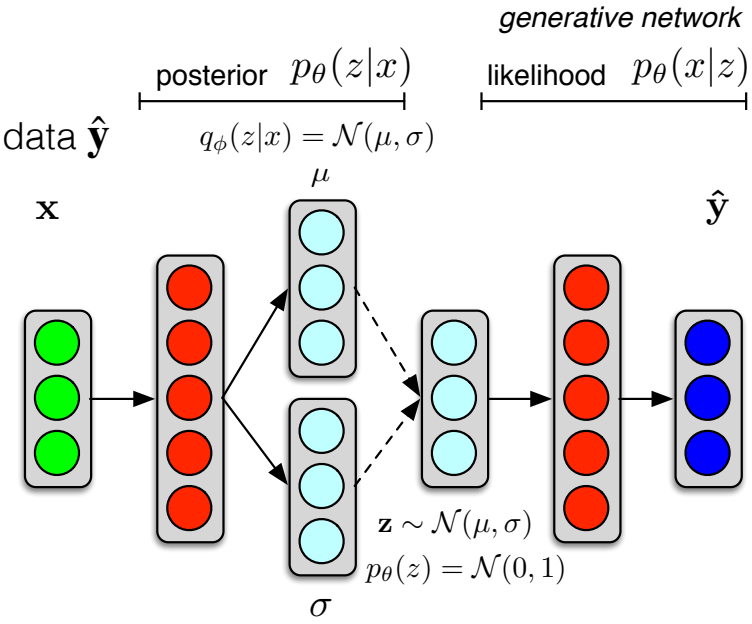
- Two sub-networks:
  - **Encoder** ϕ_e
    - projects the input data $\mathbf{x} \in \mathbb{R}^M$ in a latent representation: $\mathbf{z} = \phi_e(\mathbf{x}) \in \mathbb{R}^d \; (d << M)$
  - **Decoder** ϕ_d
    - attempts to reconstruct the input data from the latent representation $\hat{y} = \phi_d(\mathbf{z})$
  - ϕ_e and ϕ_d can be any type of network (MLP, CNN, RNN, ...)
- **Training**:
  - minimizing a MSE $\arg\min_{\phi_e,\phi_d} \|\mathbf{x} - (\phi_d \circ \phi_e(\mathbf{x}))\|^2$
- Often used for feature learning, compression, ...

- **Variations:**
  - Denoising AE
  - Sparse AE
  - Contractive AE



Variational Auto-Encoder (VAE)

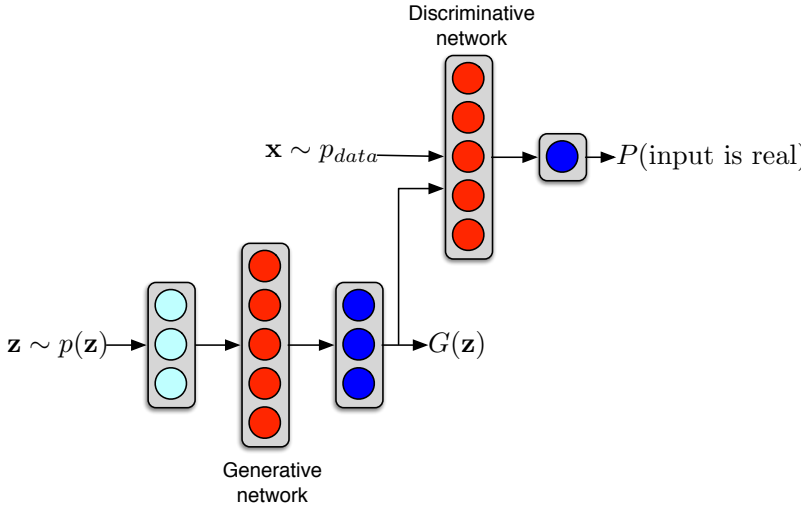
- Generative model:
  - sample points $\mathbf{z}$ in the latent space to generate new data $\hat{\mathbf{y}}$
  - most popular form of AE for generation
- **Encoder**
  - models the posterior $p_\theta(\mathbf{z}\,|\,\mathbf{x})$
- **Decoder** (generative network)
  - models the likelihood $p_\theta(\mathbf{x}\,|\,\mathbf{z})$
- Problem:
  - $p_\theta(\mathbf{z}\,|\,\mathbf{x})$ is intractable,
- Solution:
  - approximate it with $q_\phi(\mathbf{z}\,|\,\mathbf{x})$ (variational Bayesian approach) which is set to a Gaussian distribution μ, Σ (outputs of the encoder)
- **Training**:
  - minimize the KL-divergence between $q_\phi(\mathbf{z}\,|\,\mathbf{x})$ and $p_\theta(\mathbf{z}\,|\,\mathbf{x})$ equivalent to maximize an ELBO criteria
    - need a prior $p_\theta(\mathbf{z})$ which is set to $\mathcal{N}(0,1)$
  - maximize $\mathbb{E}_q[\log(x\,|\,z)]$ using Monte-Carlo ($z \sim q_\phi(z\,|\,x)$)



Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In Proc. of ICLR (International Conference on Learning Representations), 2014.

Generative Adversarial Network (GAN)

- GAN only contains the decoder part of an AE here named "**Generator**" $G(.)$
- $\mathbf{z}$ is explicitly sampled from a chosen distribution $p(z)$
- Generator $G(z)$
  - learn to generate data that look real, i.e. the distribution of the generated data p_G should look similar to the ones of real data p_{data}
- How ?
  - define a second network, the "**Discriminator**" $D(.)$ which goal is to discriminate between real and fake
- **Training** ?
  - D and G are trained in turn using a minmax optimisation
    - for G fixed, D is trained to recognized real data from fake ones
    - for D fixed, G is trained to fool D



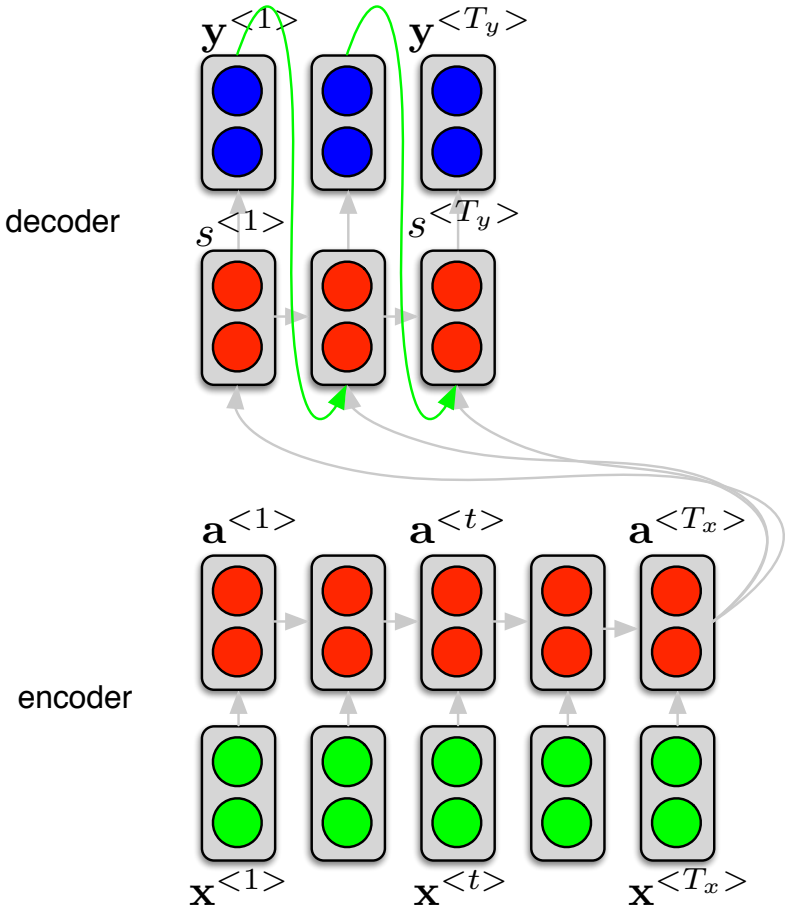
Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde- Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Advances in neural information processing systems, pages 2672–2680, 2014.

Meta-architectures

Encoder/Decoder (ED) or Sequence-to-Sequence

- encode an input sequence $\{\mathbf{x}^{<1>}\ldots\mathbf{x}^{<t>}\ldots\mathbf{x}^{<T_x>}\}$ into $\mathbf{z}$ which then serves as initialization (to condition) for decoding a sequence $\{\mathbf{y}^{<1>}\ldots\mathbf{x}^{<\tau>}\ldots\mathbf{y}^{<T_y>}\}$ into another domain

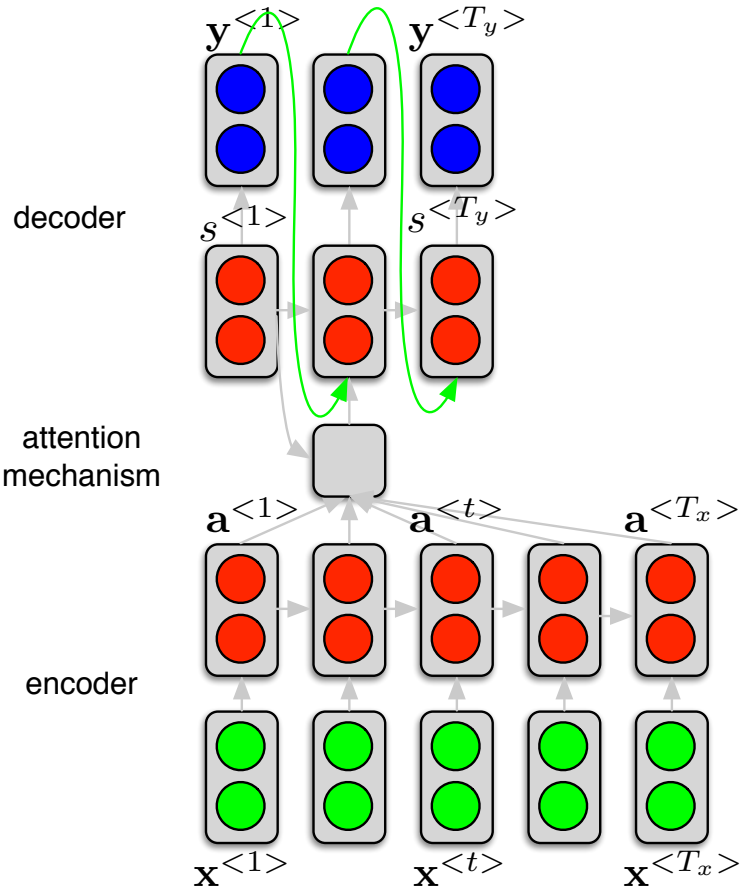
- **Usage**: machine-translation, image captioning, music style transfer



- Kyunghyun Cho, Bart Van Merri¨enboer, Caglar Gulcehre, Dzmitry Bah- danau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078, 2014.
- Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In Advances in neural information processing system pages 3104–3112, 2014.

Attention Mechanism

- Limitation of ED or Sequence-to-Sequence:
  - $\mathbf{z} = a^{<T_x>}$ is supposed to represent the whole (potentially very long) input sequence
- Attention mechanism
  - mechanism to let the decoder choose at each decoding time τ the most informative times t of the encoding internal states $\mathbf{a}^{<t>}$
  - small network trained to align encoding and decoding internal states



Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural ma- chine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473, 2014.

Meta-architectures

Transformer

- get rid of complex recurrent or convolutional neural networks that include an encoder and a decoder
- a new simple network architecture based solely on attention mechanisms, dispensing with recurrence and convolutions entirely
- still an encoder decoder
- self-attention
- positional encoding

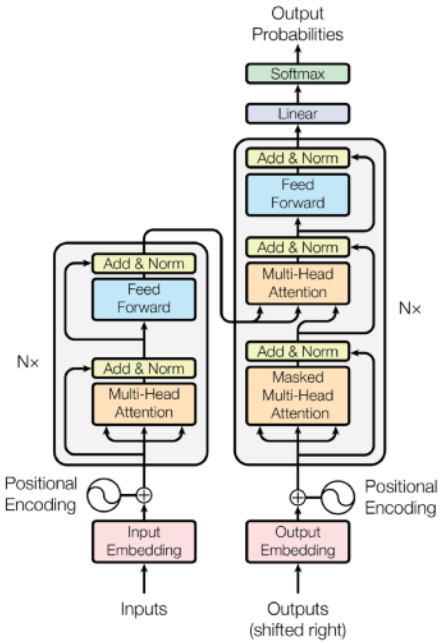
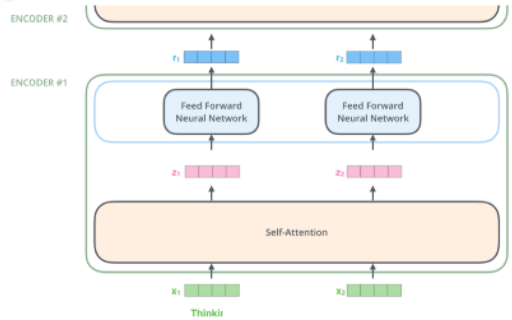
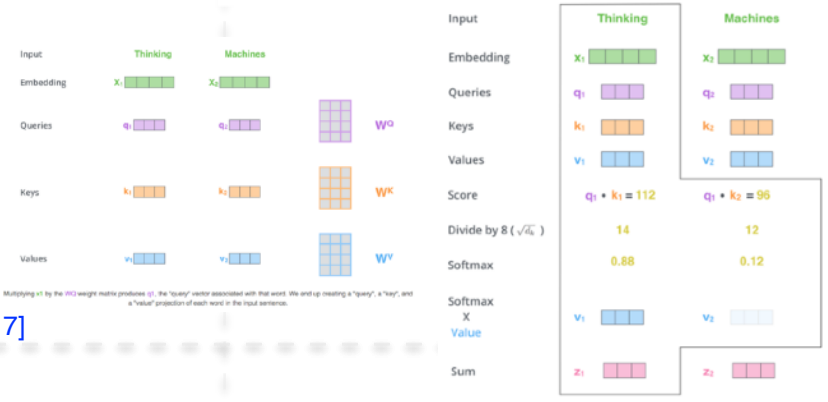
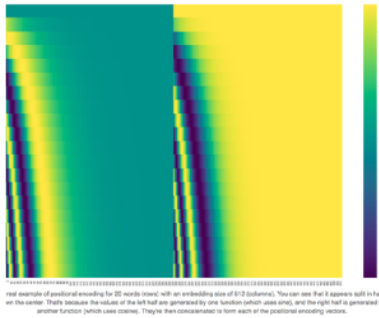


Figure 1: The Transformer - model architecture.







[Vaswani et al. "Attention Is All You Need", NIPS, 2017]
[J. Alammar "The Illustrated Transformer"]

Deep Learning reminders

Training paradigms and losses

Classification

– **Binary**

- Single output neuron (with sigmoid)
- Predicts the likelihood of positive class: $\hat{y} = p(y = 1 \,|\, x)$
- Training: minimize the Binary-Cross-Entropy (BCE):

$$\mathscr{L}^{(i)} = -\left[y^{(i)} \log(\hat{y}^i) + (1 - y^{(i)}) \log(1 - \hat{y}^i)\right]$$

– **Multi-class: predict a class c among C mutually-exclusive classes**

- Each class $c \rightarrow$ output neuron y_c (with a softmax activation)
- Predicts the likelihood of class c: $\hat{y}_c = p(y = c \,|\, x)$
- Training: minimize the Cross-Entropy (CE):

$$\mathscr{L}^{(i)} = -\sum_c \left[y_c^{(i)} log(\hat{y}_c^i)\right]$$

– **Multi-label: predict a set of class $\{c_i\}$ among C non-mutually-exclusive classes**

- Consider each class c as an **independent** binary classification (with sigmoid): $\hat{y}_c = p(y_c = 1 \,|\, x)$
- Training: minimize the sum of the BCEs of each class c:

$$\mathscr{L}_c^{(i)} = -\left[y_c^{(i)} \log(\hat{y}_c^i) + (1 - y_c^{(i)}) \log(1 - \hat{y}_c^i)\right]$$

$$\mathscr{L}^{(i)} = \sum_c \mathscr{L}_c^{(i)}$$

Reconstruction

- Goal of the network
  - reconstruct the input data $\mathbf{x}$

- **Mean Square Error**
  - $MSE = \sum_{i=1}^{N} \|\mathbf{x} - \hat{\mathbf{y}}\|^2$

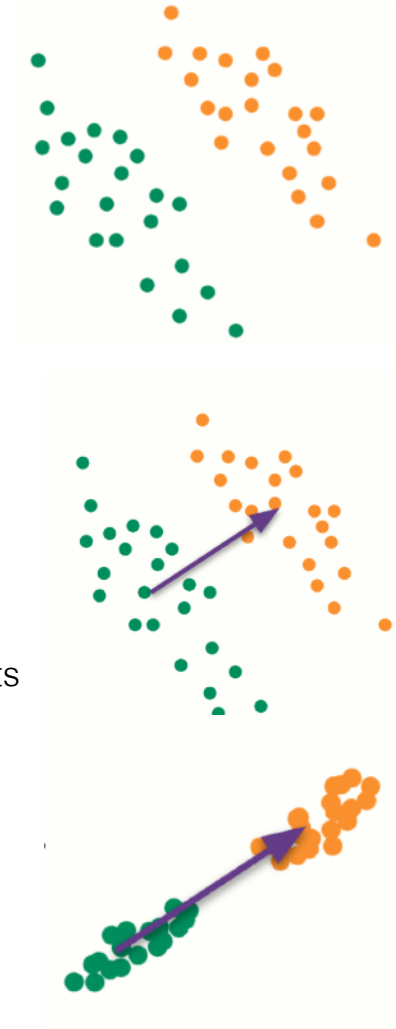
Deep Learning reminders

Training paradigms and losses

Metric learning

Metric Learning

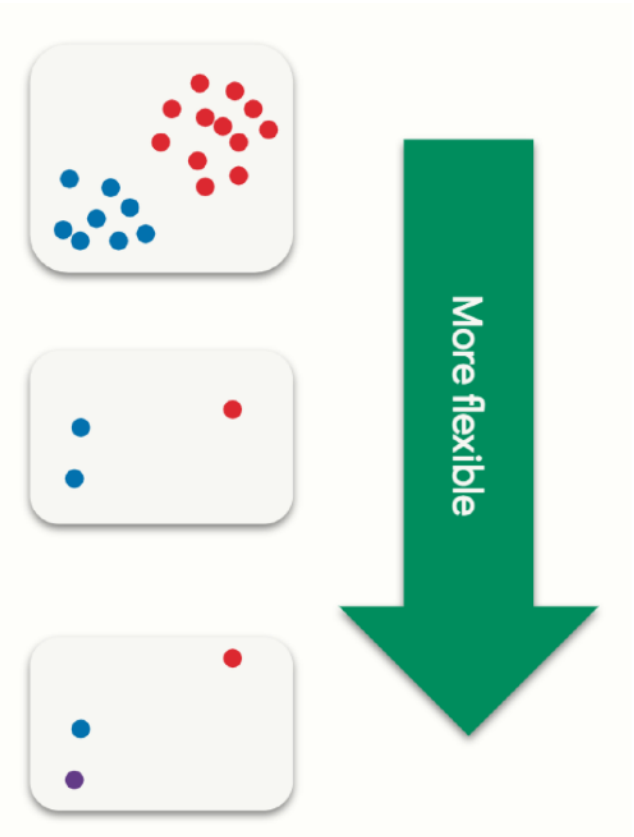
- **Distance metrics**
  - $x_1, x_2 \in \mathbb{R}^d$ two data points to be compared
  - **Euclidean distance**
    - $\|x_1 - x_2\|^2 = (x_1 - x_2)^T(x_1 - x_2) = \sum_d (x_1[d] - x_2[d])^2$
  - **Linear metric learning** seeks a projection $M \in \mathbb{R}^{DxD}$
    - $\|Mx_1 - Mx_2\|^2 = (Mx_1 - Mx_2)^T(Mx_1 - Mx_2) = (x_1 - x_2)^T M^T M(x_1 - x_2)$
    - After projection, similar points should have a small distance and dissimilar points should gave a large distance
    - Linear discriminant analysis [Fisher, 1935]
      - minimize the distance between points from the same class
      - maximize the distance distances between points from different classes



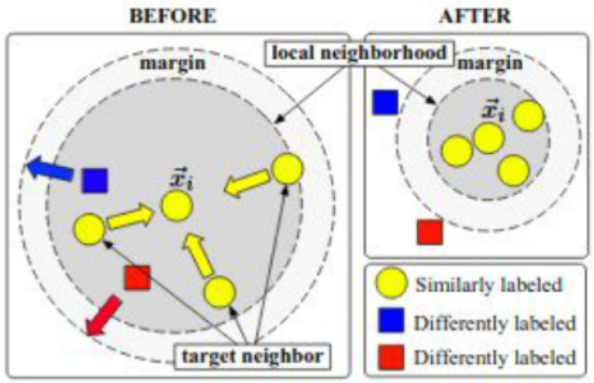
Training paradigms and losses

Metric Learning

- Different kinds of supervision
  - **Class labels**
    - (x, y)
  - **Pairwise similarity/dissimilarity**
    - $(x_1, x_2, \pm)$
  - **Relative comparisons (triplet)**
    - $(x_1, x_2, x_3) \Rightarrow d(x_1, x_2) < d(x_1, x_3)$

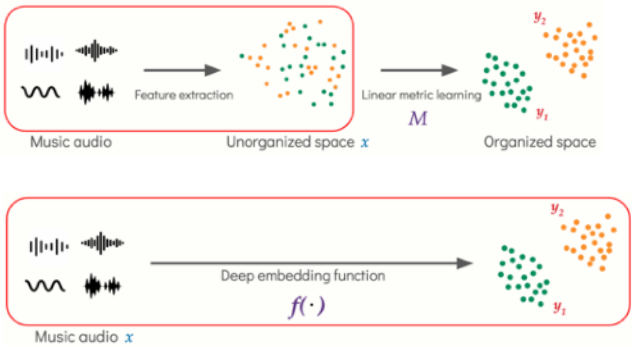


Large Margin Nearest Neighbors



- For each example x_i
  - find the k nearest "target" neighbors $\{x_j\}$ with the same label as x_i
- Each of the targets should be closer to x_i
  - than any other differently labeled point x_k by at least a margin
  - $d_{ij} + 1 \leq d_{ik} \quad \Rightarrow \quad d_{ij} + 1 - d_{ik} \leq 0$
    - $\min_M \sum_{i,j,k} \max\left(0, \quad \|Mx_i - Mx_j\|^2 - \|Mx_i - Mx_k\|^2 + 1\right)$
- Optimize M by (stochastic) gradient descent on training data

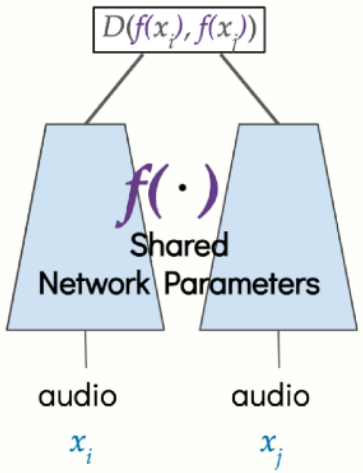
- **From linear to non-linear**
  - $D(f(x_1), f(x_2) = \|f(X_1) - f(x_2)\|^2$
  - $\min_f \sum_{i,j,k} \max\left(0, \quad \|f(x_i) - f(x_j)\|^2 - \|f(x_i) - f(x_k)\|^2 + 1\right)$



[Weinberger, Blitzer, Saul, 2006]

Metric Learning

- **Siamese Networks**
  - The same network f_θ is used to project in parallel two input data x_1 and x_2



- **Contrastive Loss**
  - i[th] labeled sample pair: $(y, x_1, x_2)^i$
    - if x_1 and x_2 are similar $\Rightarrow y^i = 0$
    - if x_1 and x_2 are dis-similar $\Rightarrow y^i = 1$
  - we define the parameterized distance function to be learned D between x_i and x_j as
    - $D_\theta^i = \|f_\theta(x_1) - f_\theta(x_2)\|^2$
  - Contrastive loss
    - minimizing $\mathscr{L}$ w.r.t. θ should result in low values of D_θ for similar pairs and high values of D_θ for dissimilar pairs

$$\mathscr{L}_\theta^i = \sum_{i=1}^{P} (1 - y^i)\mathscr{L}_S(D_\theta^i) + y^i \mathscr{L}_D(D_\theta^i)$$

$$= \sum_{i=1}^{P} (1 - y^i)\frac{1}{2}(D_\theta^i)^2 + y^i \frac{1}{2}\{\max(0, \alpha - D_\theta^i)\}^2$$

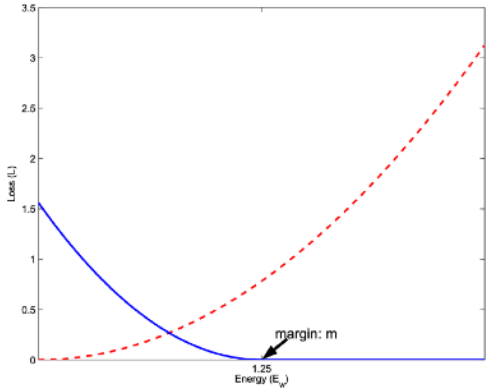


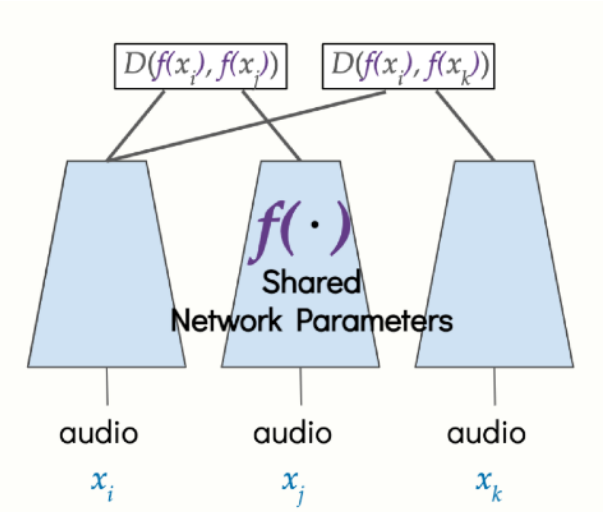
Figure 1. Graph of the loss function L against the energy D_W. The dashed (red) line is the loss function for the similar pairs and the solid (blue) line is for the dissimilar pairs.

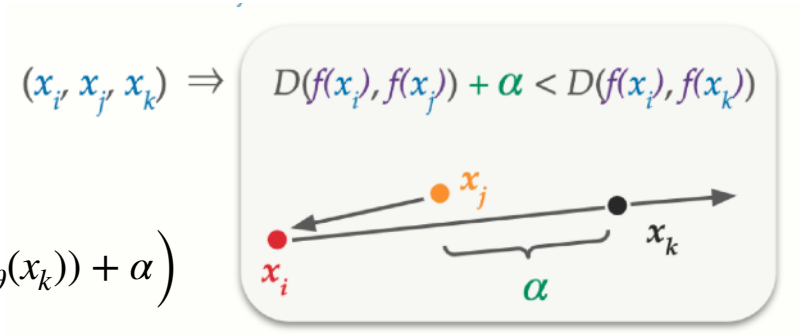
[R. Hadsell, S. Chopra, Y. LeCun, "Dimensionality Reduction by Learning an Invariant Mapping", CVPRR, 2006

Training paradigms and losses

Metric Learning

- **Triplet Loss**
  - 3 data are simultaneously considered:
    - anchor a
    - positive p (similar to a)
    - negative n (dissimilar to a)
  - Goal:
    - train the network such that $P = f_\theta(p)$ will be closer to $A = f_\theta(a)$ than $N = f_\theta(n)$ is to $A = f_\theta(a)$
  - **Minimize a triplet loss**
    - $\mathscr{L} = \max(0,\ d(A,P) + \alpha - d(A,N))$
    - α: a margin (safety) parameter
    - d: can be a simple Euclidean distance
  - **Triplet Hinge Loss**
    - $D(f_\theta(x_1) - f_\theta(x_2)) = \|f_\theta(x_1) - f_\theta(x_2)\|^2$
    - $\min_\theta \sum_{i,j,k} \max\left(0,\ D(f_\theta(x_i) - f_\theta(x_j)) - D(f_\theta(x_i) - f_\theta(x_k)) + \alpha\right)$



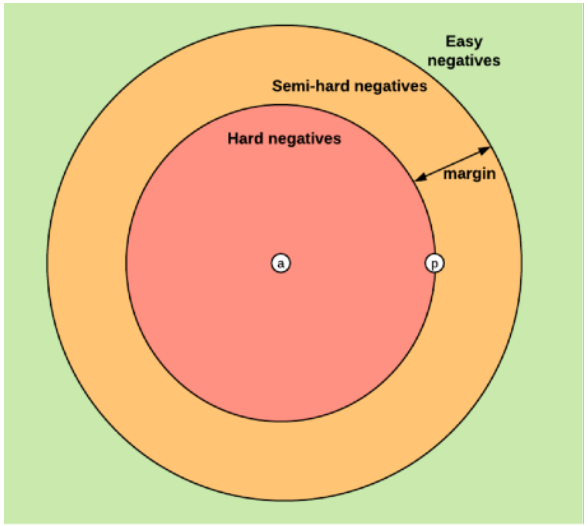


E. Hoffer, N. Ailon "Deep Metric Learning using Triplet Network", ICLR, 2015

Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In Proc. of IEEE CVPR, 2015.

Metric Learning

- **Triplet Loss $\Rightarrow$ Mining (a, p, n) triplets ?**
  - Easy negatives:
    - $d(A,P) + \alpha < d(A,N)$
  - Semi-hard negatives:
    - $d(A,P) < d(A,N) < d(A,P) + \alpha$
  - Hard negatives:
    - $d(A,N) < d(A,P)$



source: https://omoindrot.github.io/triplet-loss

Chao-Yuan Wu et al. "Sampling Matters in Deep Embedding Learning", 2017

Deep Learning reminders

Training paradigms and losses

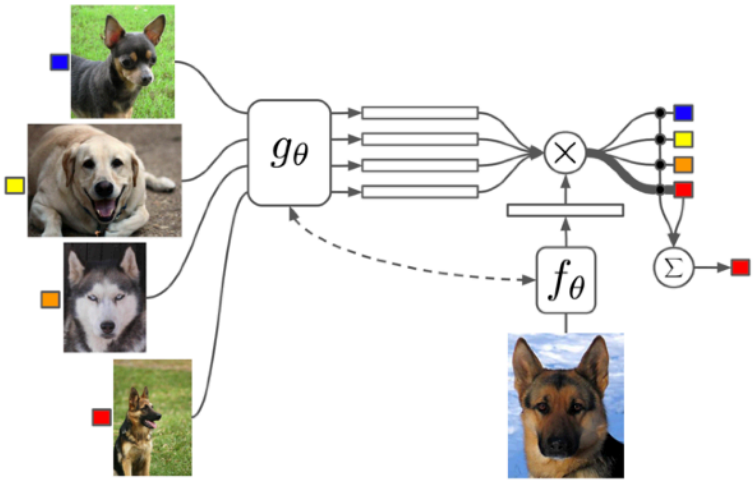
Few Shot Learning

Few-shot learning ?

– making classification or regression based on a very small number of samples.

Few-shot learning: Matching networks

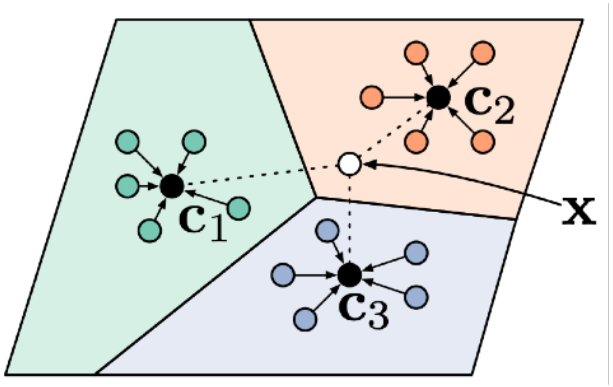
- Given
  - an input unseen example $\hat{x}$ and
  - a support set $S = \{(x_i, y_i)\}_{i=1}^{k}$,
  - appropriate label $\hat{y}$: $\arg\max_y P(y \mid \hat{x}, S)$
- The model is a network that predicts a linear combination of the support labels
  - $$\hat{y} = \sum_{i=1}^{k} a(\hat{x}, x_i)\, y_i$$
- where $a(\,.\,)$ is an "attention mechanism":
  - $$a(\hat{x}, x_i) = \frac{e^{c\left(f(\hat{x}),\, g(x_i)\right)}}{\sum_{j=1}^{k} e^{c\left(f(\hat{x}),\, g(x_j)\right)}}$$
  - with c is the cosine similarity
  - embedding functions f and g are neural networks



O. Vinyals et al., "Matching Networks for One Shot Learning", In NeurIPS, 2016

Few-shot learning: Prototypical networks

- **Main idea**:
  - there exists an embedding in which points cluster around a single prototype representation for each class
- **Prototype**
  - the mean vector of the embedded support points belonging to its class k
  - $$c_k = \frac{1}{|S_k|} \sum_{x_i \in S_k} f_\theta(x_i)$$
  - where $\mu_k \in \mathbb{R}^d$ and S_k the support set
- **distribution** over classes for a query point x
  - based on a softmax over distances to the prototypes in the embedding space
  - $$p(y = k \mid x) = \frac{e^{-d(f_\theta(x), c_k)}}{\sum_{k'} e^{-d(f_\theta(x), c_{k'})}}$$
  - Given a distance function d (Euclidean, cosine),
- Training:
  - minimize (via SGD° the negative log-probability
    - $-\log(P(y = k \mid x))$



Algorithm 1 Training episode loss computation for prototypical networks. N is the number of examples in the training set, K is the number of classes in the training set, $N_C \leq K$ is the number of classes per episode, N_S is the number of support examples per class, N_Q is the number of query examples per class. RANDOMSAMPLE(S, N) denotes a set of N elements chosen uniformly at random from set S, without replacement.

Input: Training set $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, where each $y_i \in \{1, \ldots, K\}$. $\mathcal{D}_k$ denotes the subset of $\mathcal{D}$ containing all elements $(\mathbf{x}_i, y_i)$ such that $y_i = k$.
Output: The loss J for a randomly generated training episode.

$V \leftarrow$ RANDOMSAMPLE$(\{1, \ldots, K\}, N_C)$ ▷ Select class indices for episode
for k in $\{1, \ldots, N_C\}$ **do**
  $S_k \leftarrow$ RANDOMSAMPLE$(\mathcal{D}_{V_k}, N_S)$ ▷ Select support examples
  $Q_k \leftarrow$ RANDOMSAMPLE$(\mathcal{D}_{V_k} \setminus S_k, N_Q)$ ▷ Select query examples
  $\mathbf{c}_k \leftarrow \frac{1}{N_C} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_\phi(\mathbf{x}_i)$ ▷ Compute prototype from support examples
end for
$J \leftarrow 0$ ▷ Initialize loss
for k in $\{1, \ldots, N_C\}$ **do**
  for $(\mathbf{x}, y)$ in Q_k **do**
    $J \leftarrow J + \frac{1}{N_C N_Q} \left[d(f_\phi(\mathbf{x}), \mathbf{c}_k) + \log \sum_{k'} \exp(-d(f_\phi(\mathbf{x}), \mathbf{c}_k)) \right]$ ▷ Update loss
  end for
end for

J. Snell, K. Swersky, R. Zemel, "Prototypical Networks for Few-shot Learning", In NeurIPS, 2017

Deep Learning reminders

Training paradigms and losses

Self-Supervised Learning

Self-Supervised Learning

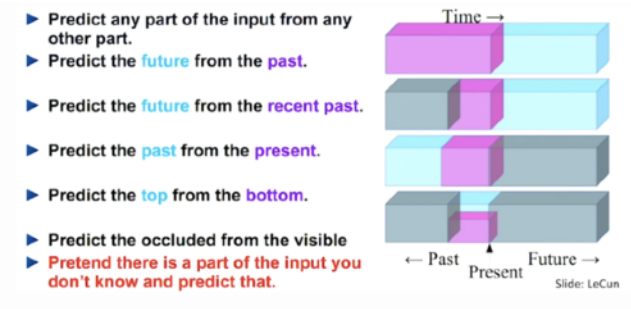
- **Supervised** learning: $\{(x_i, y_i)\}_{i=1}^{m} \rightarrow \hat{y} = f_\theta(x)$
- **Unsupervised** learning: $\{(x_i)\}_{i=1}^{m} \rightarrow \hat{y} = f_\theta(x)$
- **Self-Supervised** learning: $y_i = g(x_i)$ then $\{(x_i, y_i)\}_{i=1}^{m} \rightarrow \hat{y} = f_\theta(x)$
- A form of unsupervised learning where the data provides the supervision
  - often used to pre-train a representation f_θ (feature learning)
  - supposed to have captured the manifold in which data live and capture the overall semantic of the data
  - $y_i = g(x_i)$ is often named a **pretext tasks**
    - we are not really interested in the results for this task, it is a pretext

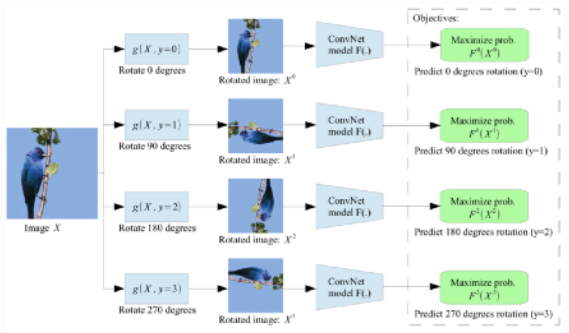
https://lilianweng.github.io/lil-log/2019/11/10/self-supervised-learning.html

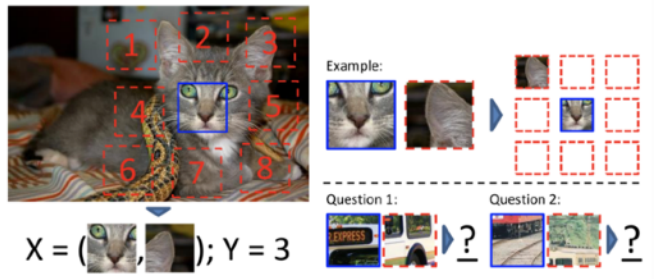
Training paradigms and losses

Self-Supervised Learning

- **... by occlusion**
  - In Natural Language Processing
    - Word2Vec
  - In Computer Vision
    - Predict the rotation of x_i
    - Predict the relative position between two random patches





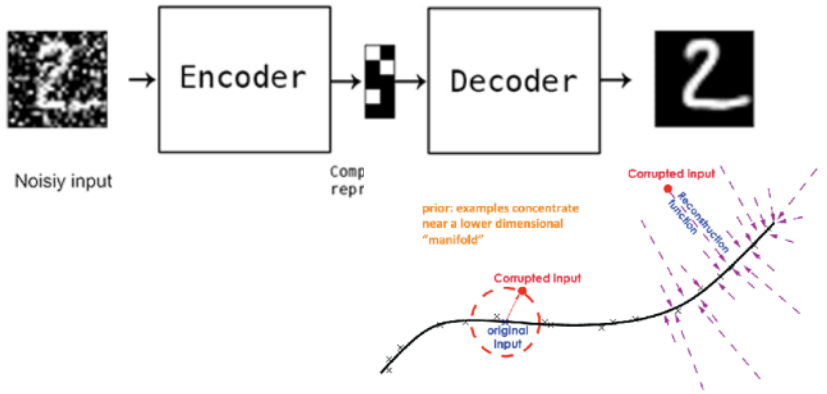


https://lilianweng.github.io/lil-log/2019/11/10/self-supervised-learning.html

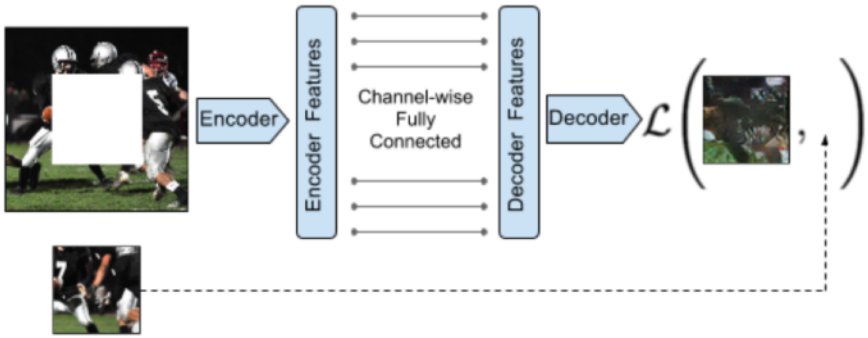
https://project.inria.fr/paiss/files/2018/07/zisserman-self-supervised.pdf

Self-Supervised Learning

- **... by generation**
  - In Computer Vision
    - Denoising auto-encoder
      - learns to recover an image from a version that is partially corrupted or has random noise

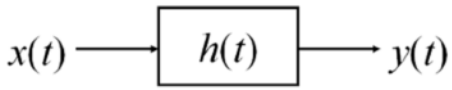


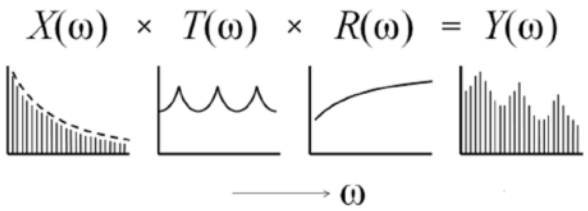
    - Context encoder
      - trained to fill in a missing piece in the image



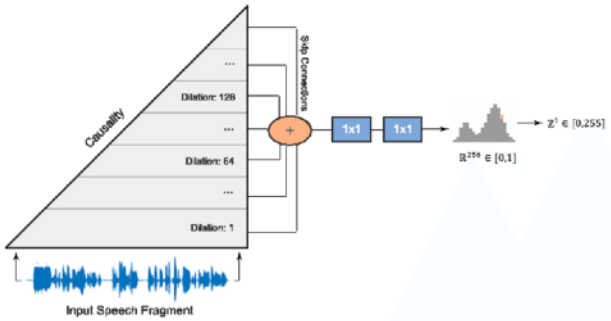
https://lilianweng.github.io/lil-log/2019/11/10/self-supervised-learning.html

2016 → Neural Autoregressive models





- Autoregressive model
  - a source/filter $x(n) = (w \circledast e)(n)$ can be associated to an autoregressive model
  - the value x_n can be predicted as a a linear combination of its P preceding values: $x_n = \sum_{p=1}^{P} a_p x_{n-p}$
- **Neural autoregressive model**
  - the linear combination is replaced by a DNN
  - a feed-forward model predicts future values from past values
- Most popular form for audio: **WaveNet** and SampleRNN
  - $p(x_n \mid x_1 \ldots x_{n-1})$ is modeled by a stack of TCNs
  - the model is trained to predict x_n which is here discretized into 256 classes (8-bits, μ-law) predicted using a softmax
  - used for speech generation
    - conditioned on side information h (speaker identity or text): $p(x_n \mid x_1 \ldots x_{n-1}, \mathbf{h})$

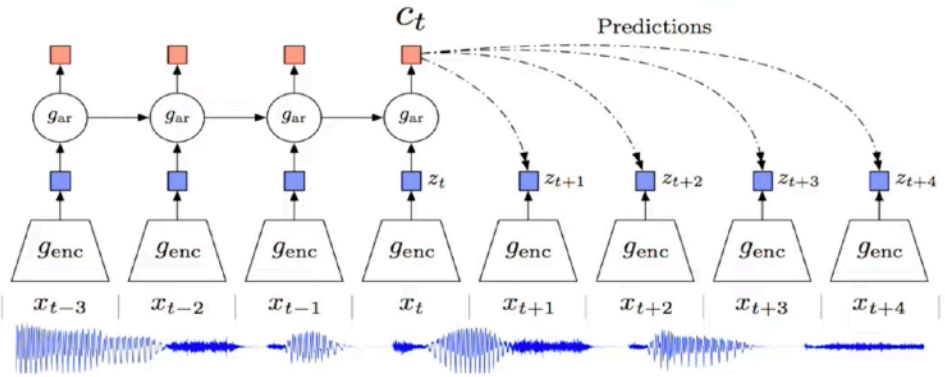


- van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., Kavukcuoglu, K.: Wavenet: A generative model for raw audio. arXiv preprint arXiv:1609.03499 (2016)
- Soroush Mehri, Kundan Kumar, Ishaan Gulrajani, Rithesh Kumar, Shubham Jain, Jose Sotelo, Aaron Courville, and Yoshua Bengio. Samplernn: An unconditional end-to-end neural audio generation model. In Proc. of ICLR (International Conference on Learning Representations), 2017

Training paradigms and losses

Contrastive Predicting Coding (CPC)

- Image
  - neighboring patches usually share spatial information locally
- Speech signals
  - neighboring times usually share similar phonemes
- CPC
  - predictions of related observations are often conditionally dependent on similar, **high-level pieces of latent information** c_t
  - complex natural data, such as images and audio, are compressed $z_t = g_{enc}(x_t)$ into a **latent embedding space** $\rightarrow$ makes the predictions $z_{t+1}, z_{t+2}, \ldots$ in the latent space conditioned on a context $c_t = g_{ar}(z_{\leq t})$;
    - g_{ar} is an autoregressive model
  - **InfoNCE loss**:
    - uses a Cross-Entropy loss to quantify how well the model can classify these future representations ("positive") from a set of unrelated "negative" examples
    - inspired by Noise Contrastive Estimation



Aaron van den Oord et al. "Representation learning with contrastive predictive coding". arXiv preprint arXiv:1807.03748, 2018.

Contrastive Predicting Coding (CPC)

- How to train ? **InfoNCE loss**
  - inspired by Noise Contrastive Estimation
  - uses a CE loss to quantify how well the model can classify these future representations ("positive") from a set of unrelated "negative" examples
  - Maximize the Mutual Information between input x and context c
    - $$I(x;c) = \sum_{x,c} p(x,c) \log \frac{p(x,c)}{p(x)p(c)} = \sum_{x,c} p(x,c) \log \frac{p(x \mid c)}{p(x)}$$
  - Rather than modeling directly the future obs. $p_k(x_{t+k} \mid c_t)$, CPC models a density function to preserve the MI between x_{t+k} and c
    - $$f_k(x_{t+k}, c_t) = \exp(z_{t+k}^T W_k c_t) \propto \frac{p(x_{t+k} \mid c_t)}{p(x_{t+k})}$$
  - Given a set of N random samples $X = \{x_1, \ldots, x_N\}$ containing
    - one positive samples $x_t \sim p(x_{t+k} \mid c_t)$ and
    - $N-1$ negative samples $x_{i \neq t} \sim p(x_{t+k})$,
    - the CE loss for classifying the positive sample correctly is
    - $$\mathcal{L}_N = -\mathbb{E}_X \left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{i=1}^N f_k(x_i, c_t)} \right]$$ looks like a softmax !

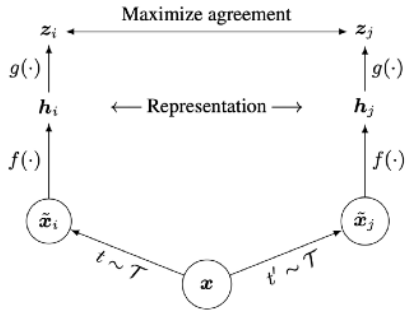
$$\begin{aligned}
\mathcal{L}_{\mathrm{N}}^{\mathrm{opt}} &= -\mathop{\mathbb{E}}_{X} \log \left[\frac{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})}}{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})} + \sum_{x_j \in X_{\mathrm{neg}}} \frac{p(x_j|c_t)}{p(x_j)}} \right] \\
&= \mathop{\mathbb{E}}_{X} \log \left[1 + \frac{p(x_{t+k})}{p(x_{t+k}|c_t)} \sum_{x_j \in X_{\mathrm{neg}}} \frac{p(x_j|c_t)}{p(x_j)} \right] \\
&\approx \mathop{\mathbb{E}}_{X} \log \left[1 + \frac{p(x_{t+k})}{p(x_{t+k}|c_t)} (N-1) \mathop{\mathbb{E}}_{x_j} \frac{p(x_j|c_t)}{p(x_j)} \right] \\
&= \mathop{\mathbb{E}}_{X} \log \left[1 + \frac{p(x_{t+k})}{p(x_{t+k}|c_t)} (N-1) \right] \\
&\geq \mathop{\mathbb{E}}_{X} \log \left[\frac{p(x_{t+k})}{p(x_{t+k}|c_t)} N \right] \\
&= -I(x_{t+k}, c_t) + \log(N),
\end{aligned}$$

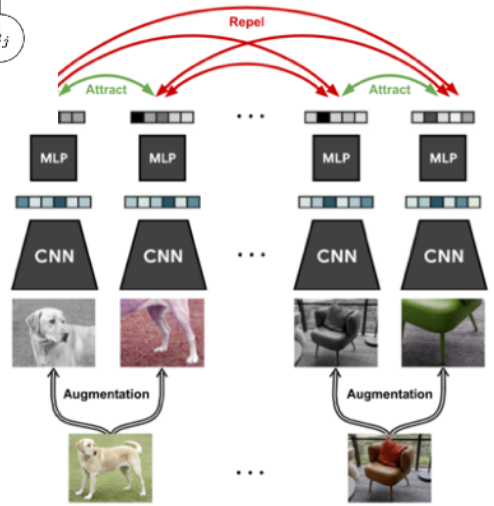
Aaron van den Oord et al. "Representation learning with contrastive predictive coding". arXiv preprint arXiv:1807.03748, 2018.

Training paradigms and losses

SimCLR

- Simple contrastive learning approach to learn strong visual representations
- strong image data augmentations
  - for each image example in the mini-batch, two augmented (but correlated!) views are taken





$$\mathcal{L}_n = log \frac{exp[S(z_n, z'_n)/\tau]}{\sum_{k \neq n} exp[S(z_n, z_k)] + \sum_k exp[S(z_n, z'_k)]}$$

$$\mathcal{L}_n{}' = log \frac{exp[S(z_n, z'_n)/\tau]}{\sum_{k \neq n} exp[S(z_n{}', z_k)] + \sum_k exp[S(z_n{}', z'_k)]}$$

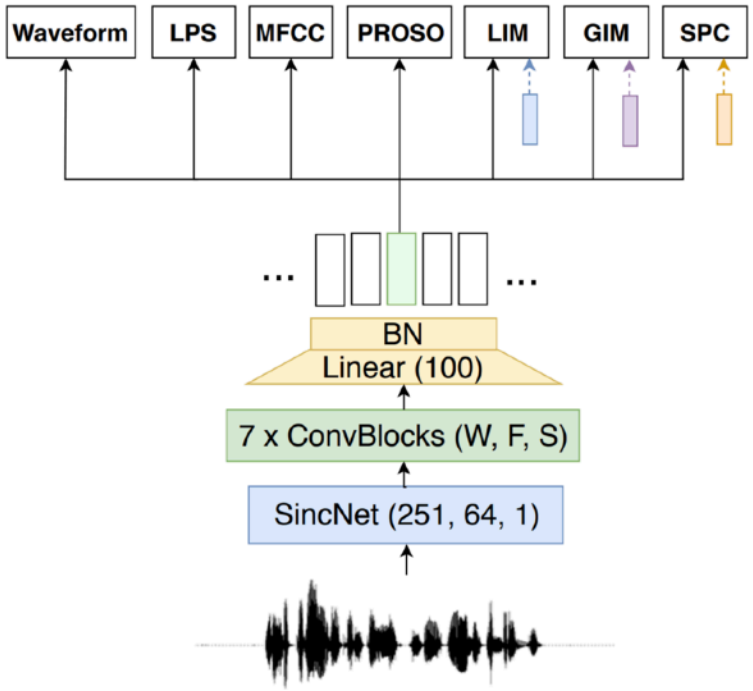
$$\mathcal{L} = \sum_n \mathcal{L}_n + \mathcal{L}_n{}'$$

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In International conference on machine learning, 1597–1607. PMLR, 2020.

Training paradigms and losses

PACE

- Optimize an encoder neural network to estimate useful representations for speech recognition (named workers, pretext tasks) such as MFCC, LPS, ...
  - each worker is composed of a single hidden layer, and either solves a regression or binary classification task
  - emphasis on learning the more expressive representations is put on the larger encoder, i.e., the encoder should learn more high-level features



Santiago Pascual, Mirco Ravanelli, Joan Serra, Antonio Bonafonte, and Yoshua Bengio. Learning problem-agnostic speech representations from multiple self-supervised tasks. arXiv preprint arXiv:1904.03416, 2019.

Training paradigms and losses

SPICE

- Task:
  - estimate the fundamental frequency in monophonic audio
- Observation:
  - pitch shift maps to a simple translation when the audio signal is analysed through the lens of the constant-Q transform (CQT)
- Self-Supervised Task:
  - feeding two shifted slices of the CQT to the same convolutional encoder, and require that the **difference in the outputs is proportional to the corresponding difference in pitch**

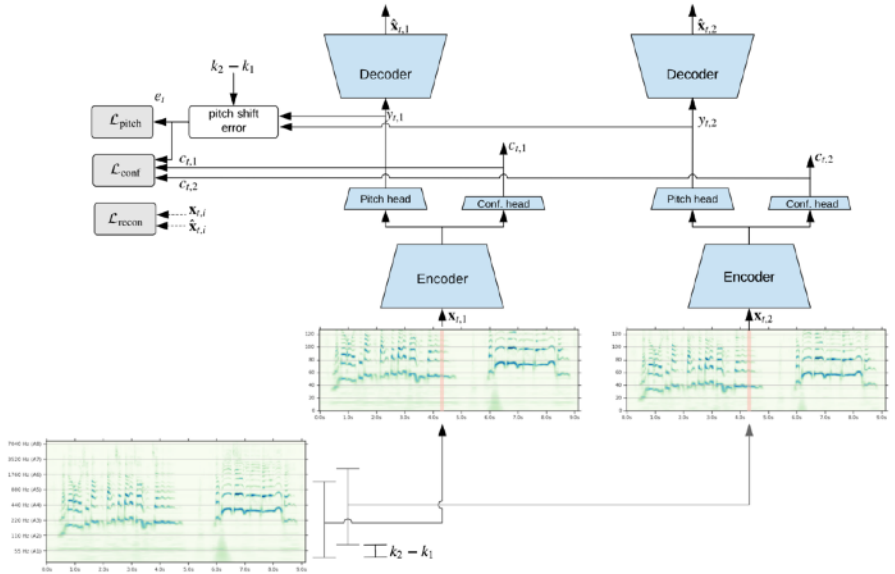


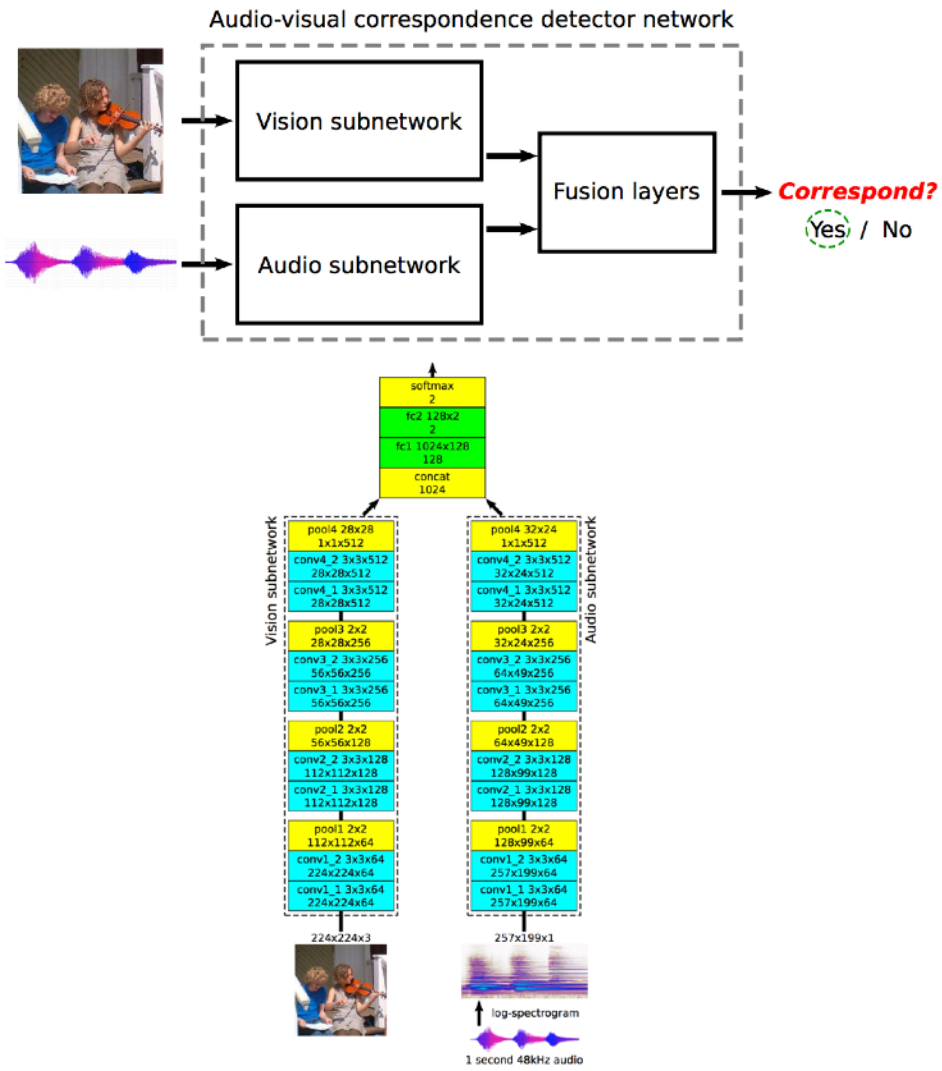
TABLE II: Evaluation results.

Model	# params	Trained on	*MIR-1k* RPA (CI 95%)	*MIR-1k* VRR	*MDB-stem-synth* RPA (CI 95%)
SWIPE	-	-	86.6%	-	90.7%
CREPE tiny	487k	many	90.7%	88.9%	93.1%
CREPE full	22.2M	many	90.1%	84.6%	92.7%
SPICE	2.38M	*SingingVoices*	90.6% ± 0.1%	86.8%	89.1% ± 0.4%
SPICE	180k	*SingingVoices*	90.4% ± 0.1%	90.5%	87.9% ± 0.9%

Beat Gfeller et al."SPICE: Self-supervised Pitch Estimation", IEEE TASLP, 2020

Audio/Visual Synchronization

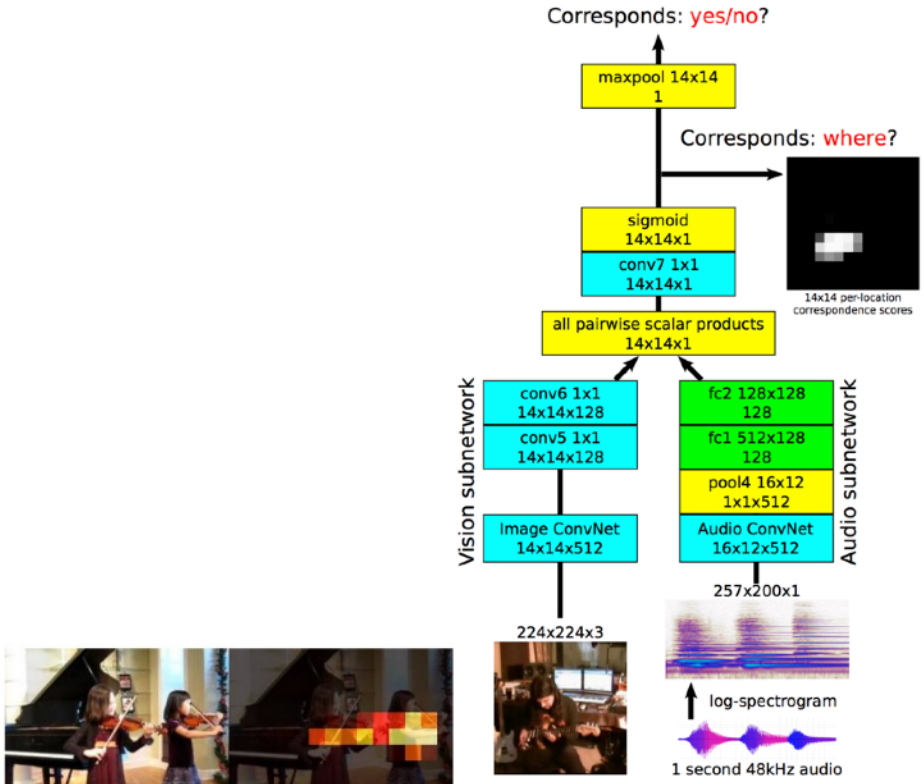
- **Look, Listen and Learn (L^3-NET)**
  - Dataset:
    - Flickr-SoundNet
    - Kinetics-Sounds (YouTube manually annotated in human actions)
  - Training from unlabelled video
  - Use only **Audio-Visual Correspondence (AVC)** as the objective function
    - positive pairs= taken at the same time from the same video



R. Arandjelovic and A. Zisserman. Look, listen and learn. In Proc. of IEEE ICCV, 2017.

Audio/Visual Synchronization

- **AVE-Net**
  - Dataset
    - AudioSet-Instruments
  - Score computed as a function of the Euclidean distance between the normalized V and A embeddings
    - enforce feature alignment
- **Applications**:
  - **querying across modalities:**
    - image ⇒ sounds ?
  - **localizing objects that sound:**
    - local region-level image descriptors are extracted on a spatial grid and a similarity score is computed between the audio embedding and each of the vision descriptors.



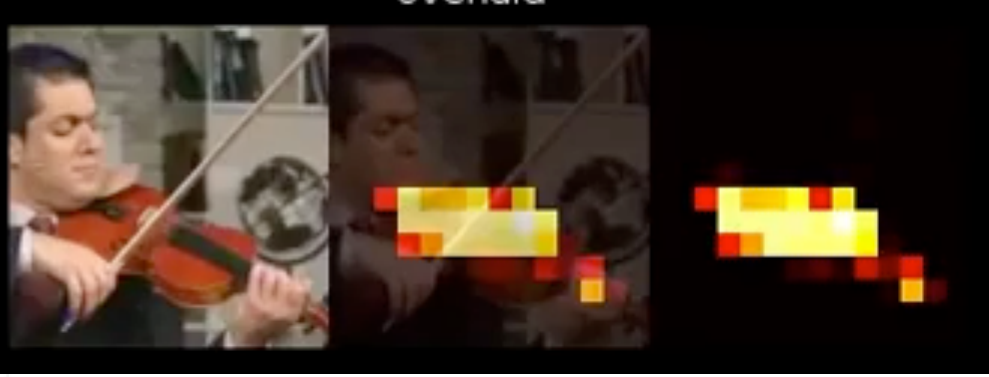
R. Arandjelovic and A. Zisserman. Objects that sound. In ECCV, 2018.

N. Mor, L. Wolf, A. Polyak, and Y. Taigman.
"A universal music translation network" in ICLR 2019

Objects that Sound

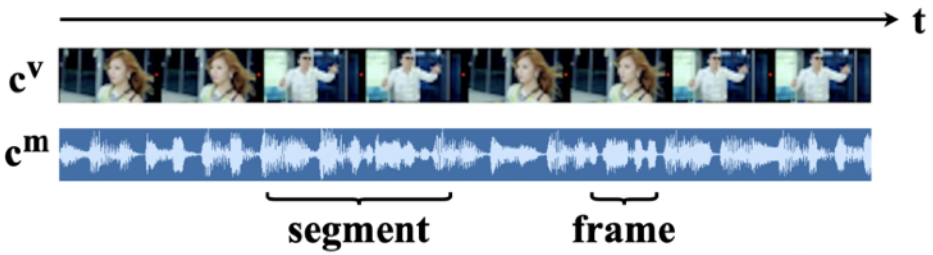
Relja Arandjelović[1] , Andrew Zisserman[1,2]
[1]DeepMind [2]University of Oxford

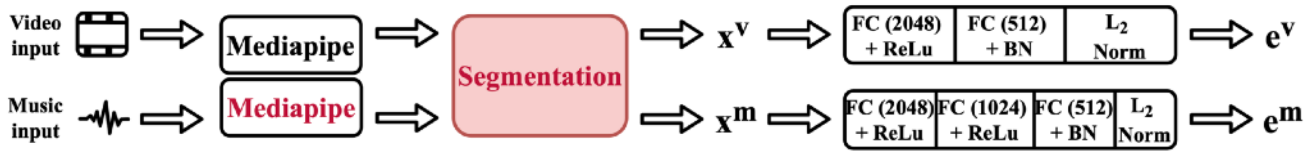
Frames are processed completely independently, motion information is not used, and there is no temporal smoothing

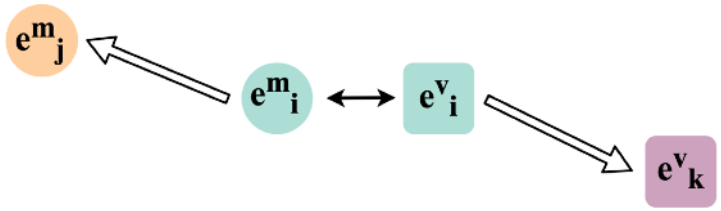


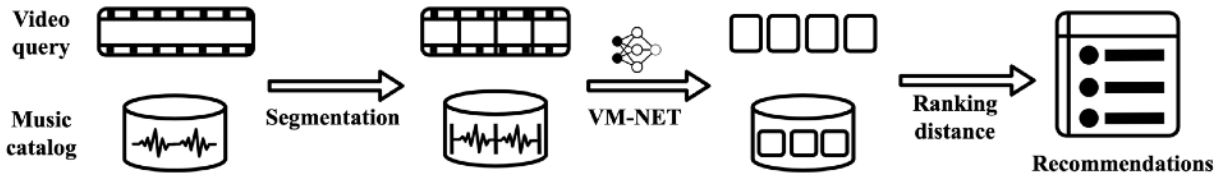
Audio/Visual Synchronization

- Goal:
  - learn a joint projection of the audio and the video content of Music-Video-Clip,
  - at the segment level: matching data are from the same segment
- **Usage**:
  - recommend a video given a music-audio (automatic Music Video Clip generation)








Video query
Music catalog
Segmentation
VM-NET
Ranking distance
Recommendations


Video Query v_q
Top 3 recommended music tracks
Histogram of $\delta(v_q, m_*)$
δ = -3.71
δ = -3.61
δ = -3.59
Rank(m_q) = 5
-3.8 , -3.42
1.6
Pop_Rock_96
Pop_Rock_39
Country_54
Country_87
δ = -4.18
δ = -3.92
δ = -3.63
Rank(m_q) = 35
-4.2
-2.55
1.4
Dubstep_7
Dubstep_34
Dubstep_33
Dubstep_45
δ = -4.04
δ = -3.85
δ = -3.58
Rank(m_q) = 1
-4.04
2.4
Reggae_26
Reggae_26
Reggae_79
Reggae_83

L. Pretet, G. Richard, C. Souchier, and G. Peeters.
"Video-to-music recommendation using ..." submitted to IEEE Trans. Multimedia 2021

Music-Video Recommendation using Temporal Alignment of Segments Demo video

Laure PRÉTET - Gaël RICHARD - Geoffroy PEETERS
LTCI, Télécom Paris, IP Paris, France
Bridge.audio, Paris, France

Deep Learning reminders

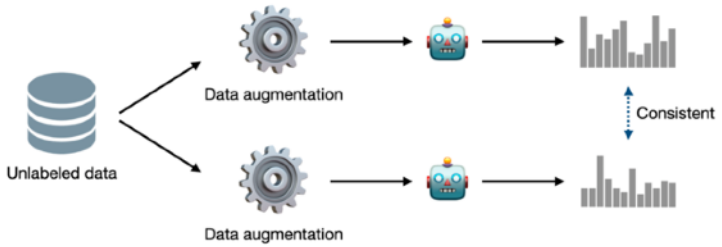
Training paradigms and losses

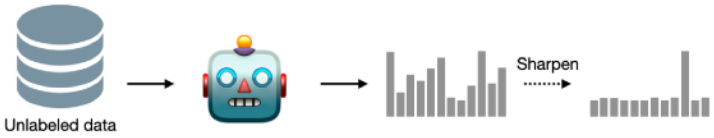
Semi-Supervised Learning

Semi-Supervised Learning

- When data=
  - small set of labeled data $\mathcal{X} = \{(x_i, y_i)\}_{i=1}^{m}$,
  - large set of un-labeled $\mathcal{U} = \{x'_i\}_{i'=1}^{m'}$
- Set of methods to combine
  - Supervised Learning
  - Unsupervised/ Self-Supervised Learning
  - $\mathcal{L} = \mathcal{L}_s + \mu \mathcal{L}_u$
- Criteria for $\mathcal{L}_u$?
  - **Pseudo-labels**
    - Train f_θ on $\mathcal{X}$, then create artificial labels for $\mathcal{U}$
  - **Consistency training**
    - minimize the difference between the prediction on x and $Augment(x)$
    - $Consistency = \|p(y|x;\theta) - p(y|Augment(x);\theta)\|$

      Virtual Adversarial Training
  - **Entropy minimization**
    - the classifier should not take decision with a large Entropy



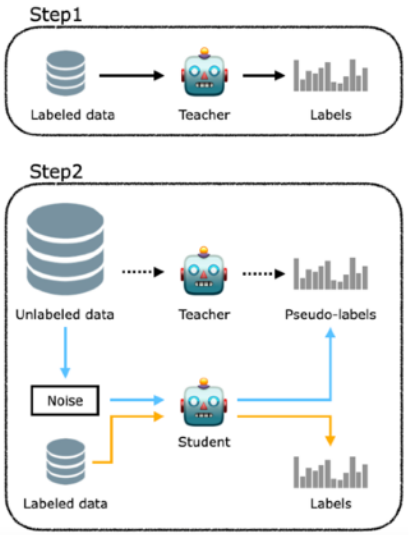


https://music-classification.github.io/tutorial/part5_beyond/methods.html
https://lilianweng.github.io/lil-log/2021/12/05/semi-supervised-learning.html

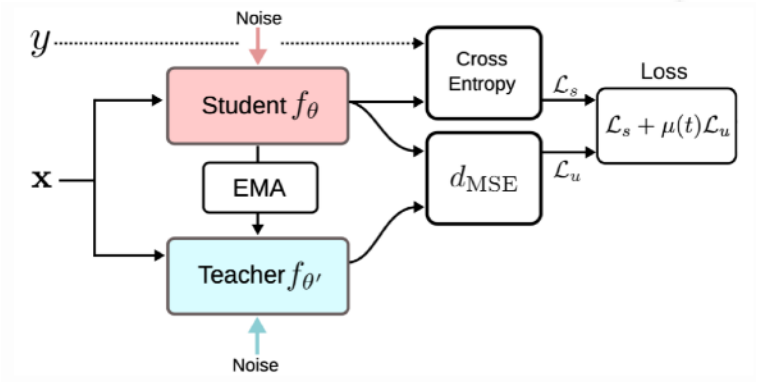
Training paradigms and losses

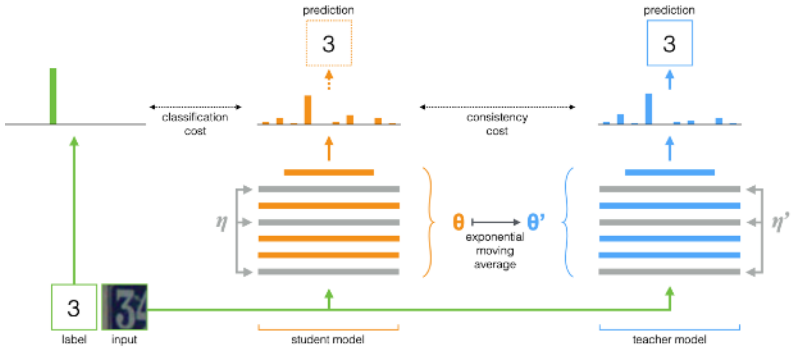
Semi-Supervised Learning

- Noisy student training
  - Student learn on pseudo-labels with noise



- Mean teacher
  - student model θ
  - teacher model (Exponential Moving Average) $\theta' \leftarrow \beta\theta' + (1-\beta)\theta$





Qizhe Xie et al. Self-training with noisy student improves imagenet classification. 2020

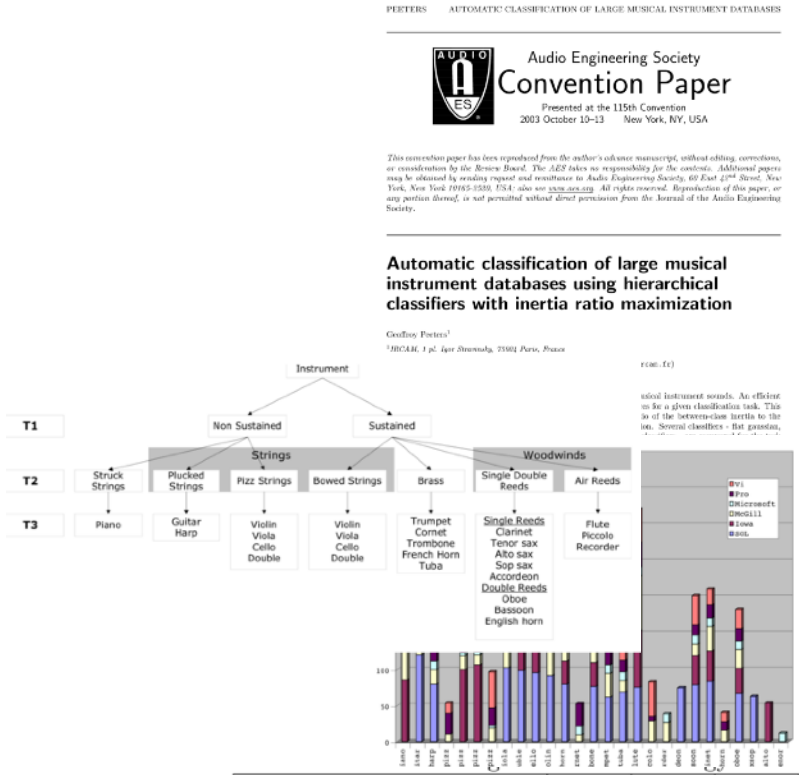
A. Tarvainen et al. Mean teachers are better role models . In NIPS, 2017

Music classification

Music classification

Instrument Classification

- Which task ?
  - single label classification, global
- Which audio features ?
  - very large set of audio features
  - automatic feature selection (IRMFSP)
  - feature transform (BoxCox, LDA)
- Which ML algorithm ?
  - flat/hierarchical Gaussian, KNN, Decision Tree
- Which datasets ?
  - 6 independent datasets (unbalanced)
- Which protocol ?
  - Cross-dataset evaluation
- Which evaluation measure ?
  - mean-over-class-Recall (balanced accuracy)


PEETERS AUTOMATIC CLASSIFICATION OF LARGE MUSICAL INSTRUMENT DATABASES

Audio Engineering Society
Convention Paper
Presented at the 115th Convention
2003 October 10–13 New York, NY, USA

Automatic classification of large musical instrument databases using hierarchical classifiers with inertia ratio maximization

Geoffroy Peeters[1]

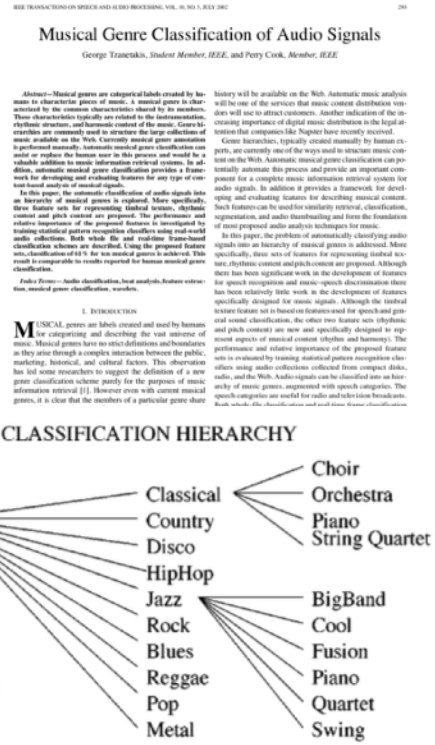
	T1	T2	T3
F-GC	98	78	55
F-GC (BC+LDA)	99	81	54
F-KNN (K=10, LDA)	99	77	51
H-GC	98	80	57
H-GC (BC+LDA)	99	85	64
H-KNN (K=10, LDA)	99	84	64
BERT	95	65	42
C4.5.		65	48
PART		71	42

G. Peeters. Automatic classification of large musical instrument databases using hierarchical classifiers with inertia ratio maximization. In Proc. of AES 115th Convention, New York, NY, USA, 2003.

Music classification

Genre classification

- Which task ?
  - single label classification, global
- Which audio features ?
  - timbral texture features
    - spectral centroid, spectral roll-off, spectral flux, time-domain zero-crossing + MFCC→ texture windows
  - rhythm content features
    - beat histogram → A0, A1, RA, P1, P1, SUM
  - pitch content features
    - enhanced autocorrelation function → FA0, UP0, FP0, IP01, S
- Which ML algorithm ?
  - simple-Gaussian (SG), GMM, KNN → weka
- Which datasets ?
  - GTZAN
- Which protocol ?
  - Ten-fold cross-validation
- Which evaluation measure ?
  - Classification accuracy


Musical Genre Classification of Audio Signals

George Tzanetakis, *Student Member, IEEE*, and Perry Cook, *Member, IEEE*

AUDIO CLASSIFICATION HIERARCHY

	Genres(10)	Classical(4)	Jazz(6)
Random	10	25	16
RT GS	44 ± 2	61 ± 3	53 ± 4
GS	59 ± 4	77 ± 6	61 ± 8
GMM(2)	60 ± 4	81 ± 5	66 ± 7
GMM(3)	61 ± 4	88 ± 4	68 ± 7
GMM(4)	61 ± 4	88 ± 5	62 ± 6
GMM(5)	61 ± 4	88 ± 5	59 ± 6
KNN(1)	59 ± 4	77 ± 7	57 ± 6
KNN(3)	60 ± 4	78 ± 6	58 ± 7
KNN(5)	56 ± 3	70 ± 6	56 ± 6

G. Tzanetakis and P. Cook. Musical genre classification of audio signals. Speech and Audio Processing, IEEE Transactions on, 10(5):293–302, 2002.

Music classification

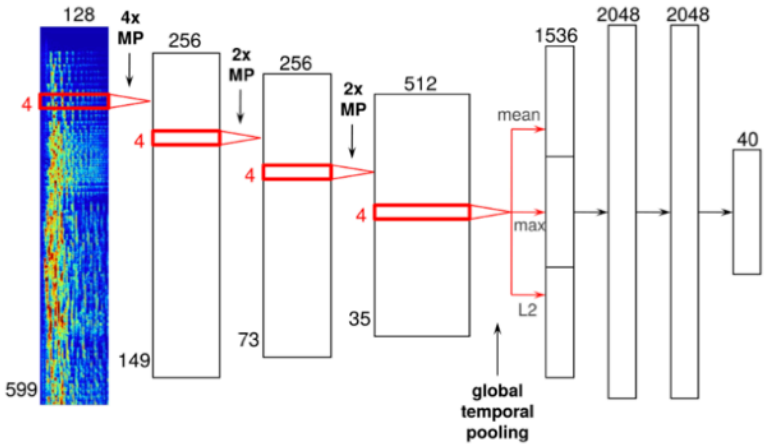
Genre classification (1D-Convolution approach)

- Which task ?
  - regression model to predict the **latent representations** of songs that were obtained from a collaborative filtering model
- Which audio features ?
  - Log-Mel-Spectrograms (599 frames and 128 frequency bins)
- Which ML algorithm ?
  - 1D-CNN → Theano
- Which datasets ?
  - Spotify
- Which protocol ?
  -
- Which evaluation measure ?
  - MSE

This summer, I'm interning at Spotify in New York City, where I'm working on content-based music recommendation using convolutional neural networks. In this post, I'll explain my approach and show some preliminary results.

1. Overview

This is going to be a long post, so here's an overview of the different sections. If you want to skip ahead, just click the section title to go there.

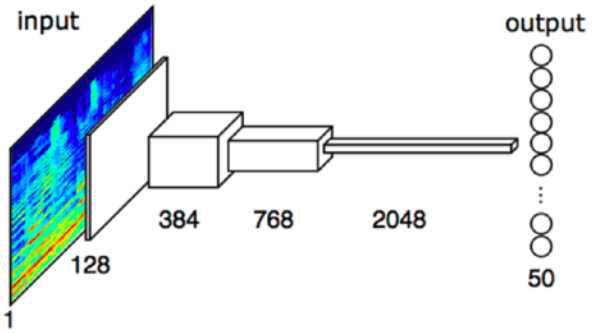


S. Dieleman. Recommending music on spotify with deep learning. Technical report, http://benanne.github.io/2014/08/05/spotify-cnns.html , 2014.
https://papers.nips.cc/paper/2013/file/b3ba8f1bee1238a2f37603d90b58898d-Paper.pdf

Music classification

Genre classification (VGG-Net approach)

- Which task ?
  - multi-label, auto-tagging
- Which audio features ?
  - Log-Mel-Spectrogram (96×1366)
- Output:
  - predicts a 50 dimensional tag vector
- Which ML algorithm ?
  - VGG-CNN
- Which datasets ?
  - Magna-Tag-A-Tune
    - 25,856 clips of 29.1-s, 16 kHz-sampled mp3 files with 188 tags
  - Million-Song-Dataset
- Which protocol ?
  - train/ test set
- Which evaluation measure ?
  - AUC-ROC



FCN-4
Mel-spectrogram *(input: 96×1366×1)*
Conv 3×3×128
MP (2, 4) *(output: 48×341×128)*
Conv 3×3×384
MP (4, 5) *(output: 24×85×384)*
Conv 3×3×768
MP (3, 8) *(output: 12×21×768)*
Conv 3×3×2048
MP (4, 8) *(output: 1×1×2048)*
Output 50×1 (sigmoid)

FCN-5	FCN-6	FCN-7
Mel-spectrogram *(input: 96×1366×1)*		
Conv 3×3×128		
MP (2, 4) *(output: 48×341×128)*		
Conv 3×3×256		
MP (2, 4) *(output: 24×85×256)*		
Conv 3×3×512		
MP (2, 4) *(output: 12×21×512)*		
Conv 3×3×1024		
MP (3, 5) *(output: 4×4×1024)*		
Conv 3×3×2048		
MP (4, 4) *(output: 1×1×2048)*		
	Conv 1×1×1024	Conv 1×1×1024
		Conv 1×1×1024
Output 50×1 (sigmoid)		

	AUC
FCN-3, mel-spectrogram	.852
FCN-4, mel-spectrogram	**.894**
FCN-5, mel-spectrogram	.890
FCN-4, STFT	.846
FCN-4, MFCC	.862

Table 3. The results of the proposed architectures and input types on the MagnaTagATune Dataset

Methods	AUC
FCN-3, mel-spectrogram	.786
FCN-4, —	.808
FCN-5, —	.848
FCN-6, —	**.851**
FCN-7, —	.845

Table 5. The results from different architectures of proposed system on the Million Song Dataset

K. Choi, G. Fazekas, and M. Sandler. Automatic tagging using deep convolutional neural networks. In Proc. of ISMIR (International Society for Music Information Retrieval), New York, USA, 2016.

Genre classification (Musically-motivated CNN approach)

- Which task ?
  - classification into rhythm patterns
- Which audio features ?
  - Log-Mel-Spectrogram samples
- Which ML algorithm ?
  - CNN with carefully designed filter-shapes to represent timbre and rhythm
- Which datasets ?
  - Ballroom dataset
- Which protocol ?
  - ten-fold cross-validation
- Which evaluation measure ?
  - Accuracy

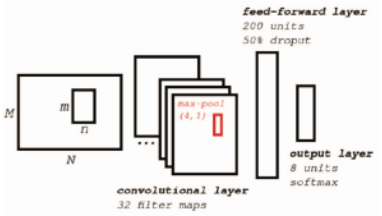


Fig. 1. Schema of the *Black-box* architecture.

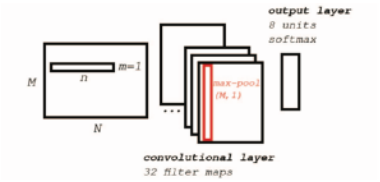


Fig. 2. Schema of the *Time* architecture.

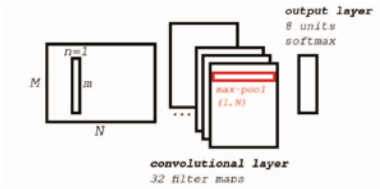
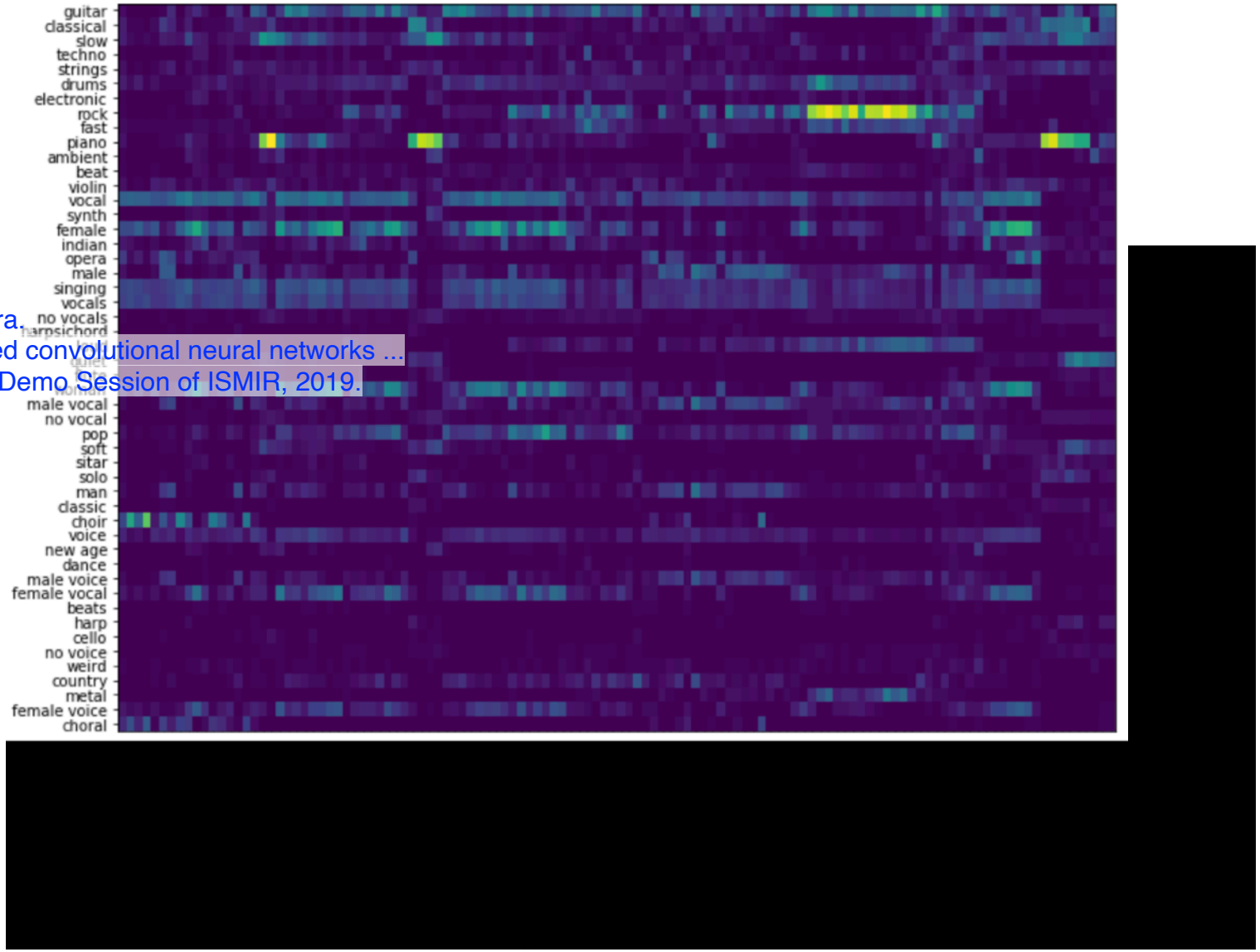


Fig. 3. Schema of the *Frequency* architecture.

Architecture	Input (M,N)	Filter shape (m,n)	# param.	Max-pool	Accuracy: mean ± std 10 cross-fold validation	Baseline
Black-box	(40,80)	(12,8)	3.275.312	(4,1)	**87.25** ± 3.39 %	93.12 % → Marchand *et al.* [11]
Black-box	(40,250)	(12,200)	2.363.440	(4,1)	82.80 ± 5.12 %	93.12 % → Marchand *et al.* [11]
Time	(40,80)	(1,60)	7.336	(40,1)	**81.79** ± 4.72 %	82.3% → Guyon *et al.* [6]
Time	(40,250)	(1,200)	19.496	(40,1)	81.52 ± 3.87 %	82.3% → Guyon *et al.* [6]
Frequency	(40,80)	(30,1)	3.816	(1,80)	59.45 ± 5.02%	15.9 % → Most probable class
Frequency	(40,80)	(32,1)	3.368	(1,80)	**59.59** ± 5.82 %	15.9 % → Most probable class
Frequency	(40,80)	(34,1)	2.920	(1,80)	58.17 ± 3.58 %	15.9 % → Most probable class
Frequency	(40,80)	(36,1)	2.472	(1,80)	57.88 ± 5.38 %	15.9 % → Most probable class
Frequency	(40,80)	(38,1)	2.024	(1,80)	57.45 ± 5.93 %	15.9 % → Most probable class
Frequency	(40,80)	(40,1)	1.576	(1,80)	52.43 ± 5.63 %	15.9 % → Most probable class
Time-Frequency	(40,80)	(1,60)-(32,1)	196.816	(40,1)-(1,80)	86.54 ± 4.29 %	93.12 % → Marchand *et al.* [11]
Time-FrequencyInit	(40,80)	(1,60)-(32,1)	196.816	(40,1)-(1,80)	**87.68** ± 4.44 %	93.12 % → Marchand *et al.* [11]

J. Pons, T. Lidy, and X. Serra. Experimenting with musically motivated convolutional neural networks. In Proc. of IEEE CBMI (International Workshop on Content-Based Multimedia Indexing), 2016.

Music classification

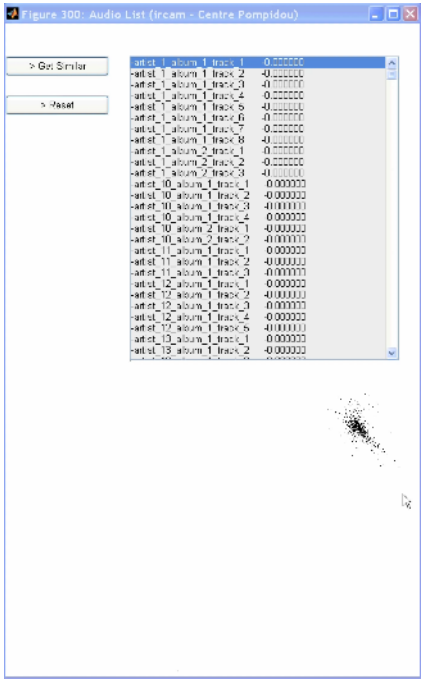


J. Pons and X. Serra.
Musicnn: Pre-trained convolutional neural networks ...
In Late-Breaking/Demo Session of ISMIR, 2019.

Music search-by-similarity (recommendation)

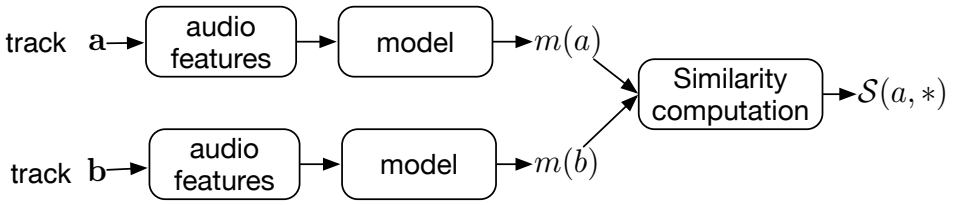
Music search-by-similarity (recommendation)

- **Goal:**
  - recommend a set of music tracks based on their acoustic-similarity (content-based) to a target track $\mathbf{a}$



- **Usage:**
  - create playlists of similar tracks to a target track $\mathbf{a}$

- **Method:**
  - (1) need to compute a similarity/distance $\mathcal{S}(a, b)$ between pairs of tracks $\mathbf{a}$ and $\mathbf{b}$
  - (2) rank the distance $\mathcal{S}(a, *)$ for every possible tracks * of the dataset, to get a playlist



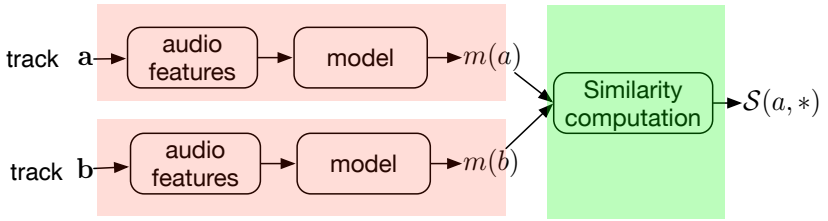
track $\mathbf{a}$ → rank $\mathcal{S}(a, *)$

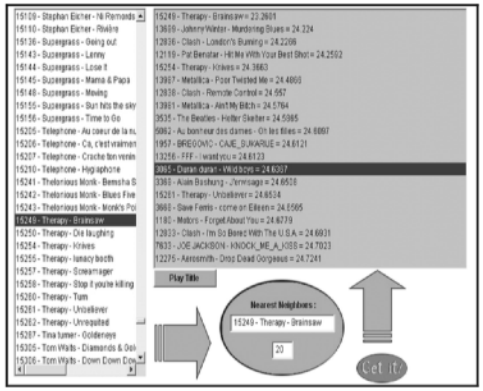
Music search-by-similarity (recommendation)

Search by similarity

- Which task ?
  - define a distance between two tracks to allow performing recommendation
- Which audio features ?
  - sequence of MFCC over time
  - the sequence is then modeled as a bag-of-MFCC using Gaussian Mixture Model
- Which ML algorithm ?
  - the similarity/distance between two tracks is done computing the Kullback-Leibler divergence between the GMM (EM-distance or Monte-Carlo sampling)
- Which datasets ?
  - personal: 17,075 popular music titles
- Which protocol ?
- Which evaluation measure ?
  - same artist, same genre
  - subjective evaluation


FINDING SONGS THAT SOUND THE SAME



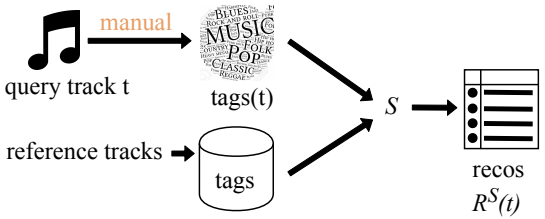


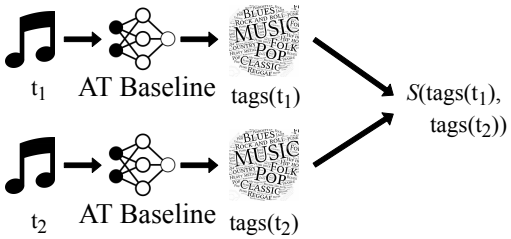
J. Aucouturier and F. Pachet. Finding songs that sound the same. In Proc. of IEEE Benelux Workshop on Model based Processing and Coding of Audio, pages 1–8, 2002.

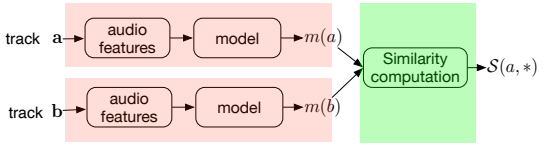
Music search-by-similarity (recommendation)

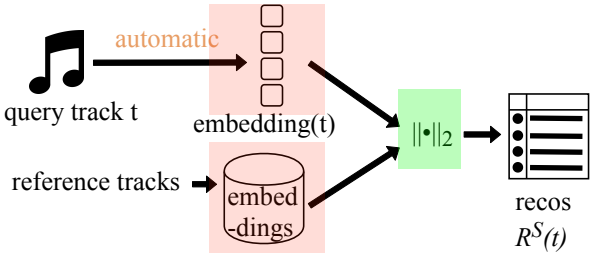
Search by similarity (triplet loss approach)

- In systems, music recommandation/similarity is based on
  - similarity of tags
  - computed as a function $\mathcal{S}$ of manually annotated tags
- Baseline idea:
  - automatically estimate the tags
- Our idea:
  - directly reproduce the ranking of $\mathcal{S}$ using only the audio





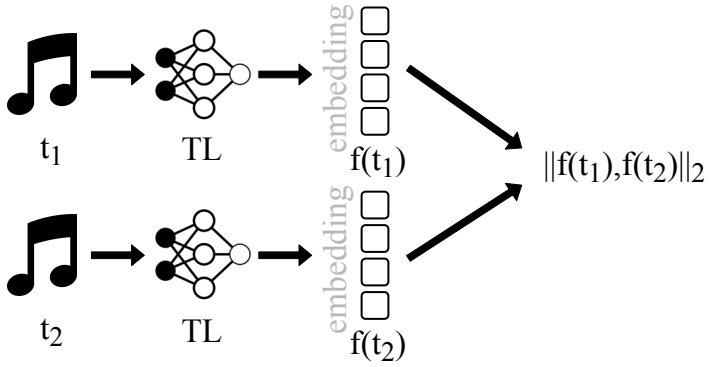




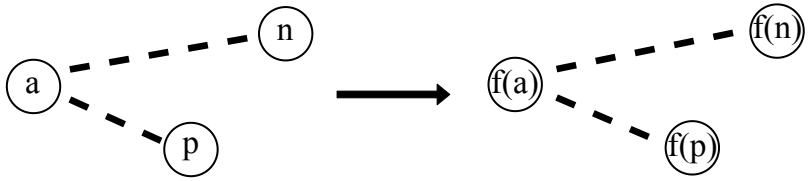
L. Pretet, G. Richard, and G. Peeters. Learning to rank music tracks using triplet loss. In Proc. of IEEE ICASSP (International Conference on Acoustics, Speech, and Signal Processing), Barcelona, Spain, May, 4–8 2020.

Search by similarity (triplet loss approach)

– How to directly reproduce the ranking of $\mathcal{S}$ using only the audio ?
  - train a DNN to learn an **embedding space** in which the Euclidean distance allows to retrieve tracks with the same ranking as with the oracle similarity function $\mathcal{S}$



– **Triplet loss approach**
  - $\mathscr{L}(a,p,n) = \max(\|f(a) - f(p)\|_2^2 - \|f(a) - f(n)\|_2^2 \pm \alpha, 0)$

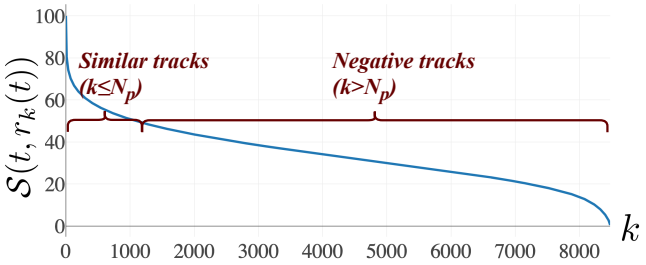


Schroff and Philibin, "FaceNet: A Unified Embedding for Face Recognition and Clustering", in Proc of CVPR, 2015.

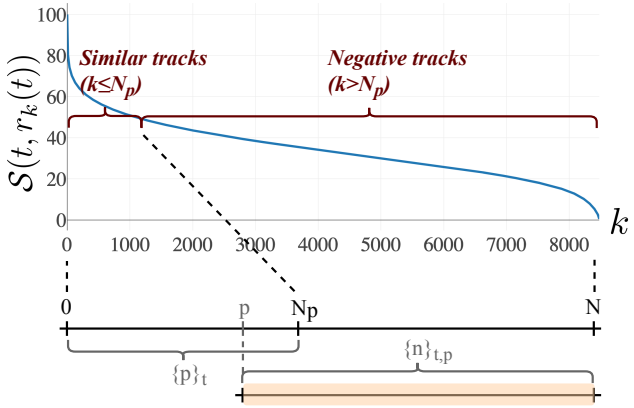
Search by similarity (triplet loss approach)

- How to select the triplets a, p, n ? → **Triplet mining**
  - usual triplet mining is based on class identity
- **Our idea: triplet mining for search-by-similarity**
  - Training data:
    - t : a target track
    - $R^{\mathcal{S}}(t) = [r_1(t), \ldots, r_{N-1}(t)]$: the associated ground-truth ranked list, given by $\mathcal{S}$
  - We define training triplets by choosing a track t as the *anchor* and by mining a *positive* and a *negative* element from $R^{\mathcal{S}}(t)$
  - Valid triplets: $(a, p, n) = (t, r_i(t), r_j(t))$ with $i < j$

Selecting the p:
only consider N_p first tracks of the reference ranked list $R^{\mathcal{S}}(t)$

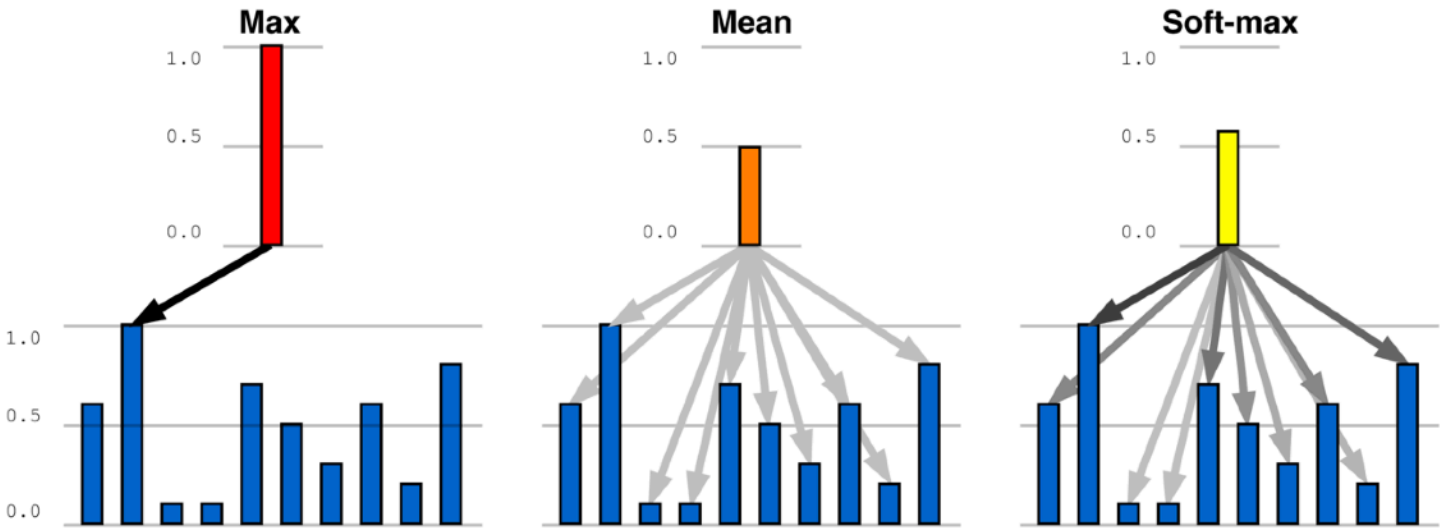


Selecting the n:
3 strategies to select the N_n negative examples for a given anchor-positive pair $(t, r_i(t))$



Search by similarity (triplet loss approach)

Adaptive pooling operators:
automatically find the best among max, mean, soft-max pooling

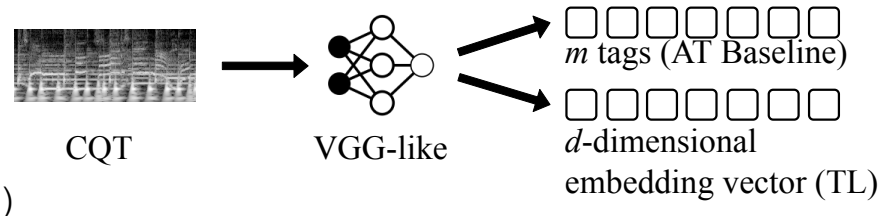


McFee et al.,"Adaptive pooling operators for weakly labeled sound event detection", TASLP, 2018.

Music search-by-similarity (recommendation)

Search by similarity (triplet loss approach)

- **Model**
  - VGG-Net ConvNet [Choi, 2016]
- **Evaluation**
  - $N = 14{,}246$ tracks (music supervision catalog)
  - Each track manually annotated into $m = 488$ tags
    - Ex: Blues, Reggae, Japanese Pop, Acoustic, Saturated, Guitar bass, Epic, Dancing, Nostalgic, Acceleration, Repetitive.
  - Function $\mathcal{S}$ specific to this dataset
  - Split into a training, validation and test sets (60%, 20% and 20% respectively)
- **Task:**
  - What we want: the 5 first tracks of the ranking: $R_5^{\mathcal{S}}(t) = [r_1^{\mathcal{S}}(t), \ldots, r_5^{\mathcal{S}}(t)]$
  - What we get: the top 20 estimated recommendations of the system: $R_k^{\hat{\mathcal{S}}}(t) = [r_1^{\hat{\mathcal{S}}}(t), \ldots, r_{20}^{\hat{\mathcal{S}}}(t)]$
- **Performance measures:**
  - MAP
  - Recall
  - Reciprocal rank (RR)
  - nDCG



Choi et al., "Automatic tagging using deep convolutional neural networks", in Proc. of ISMIR, 2016

Music search-by-similarity (recommendation)

Search by similarity (triplet loss approach)

Triplet Loss systems perform better than the Auto-Tagger baseline

Model	MAP@20	Recall@20	RR@20	nDCG@20
AT Baseline	4.50 ± 0.34	12.57 ± 0.66	15.62 ± 1.07	11.30 ± 0.69
TL Neighbors	**5.58 ± 0.41**	**12.73 ± 0.67**	**19.18 ± 1.23**	**13.41 ± 0.80**

The Distance-based mining strategy seems to give the best results

Model	MAP@20	Recall@20	RR@20	nDCG@20
TL Neighbors	5.58 ± 0.41	12.73 ± 0.67	19.18 ± 1.23	13.41 ± 0.80
TL Random	5.39 ± 0.38	15.01 ± 0.70	17.86 ± 1.12	13.50 ± 0.76
TL Distance-based	**5.98 ± 0.40**	**15.79 ± 0.73**	**19.89 ± 1.19**	**14.41 ± 0.78**

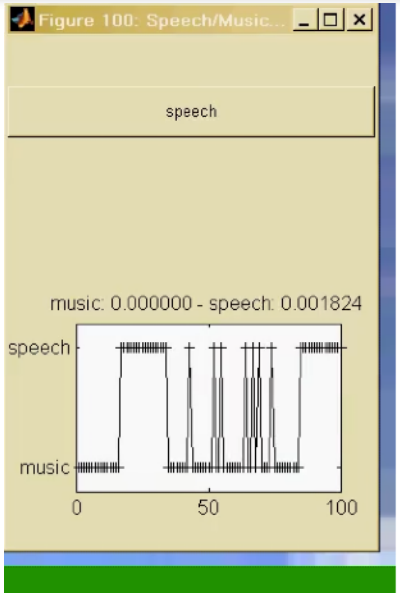
Using the Autopooling layer boosts performances

Model	MAP@20	Recall@20	RR@20	nDCG@20
TL Distance-based	5.98 ± 0.40	15.79 ± 0.73	19.89 ± 1.19	14.41 ± 0.78
TL Autopool	**7.99 ± 0.51**	**17.74 ± 0.79**	**24.68 ± 1.34**	**17.95 ± 0.92**

Speech/music segmentation

Speech/music segmentation

- **Goal:**
  - find the start/end and {speech, music} label of an audio stream into
- **Usage:**
  - segment a radio/tv show, a pod-cast, a movie
  - front-end for
    - speech recognition
    - music identification
    - music recognition
- **Methods:**
  - **A. first label each frames and then segment according to label changes**
  - B. first detect segments and then label each segments
  - C. perform both jointly (the HMM approach)
  - same techniques for singing/instrumental, ...



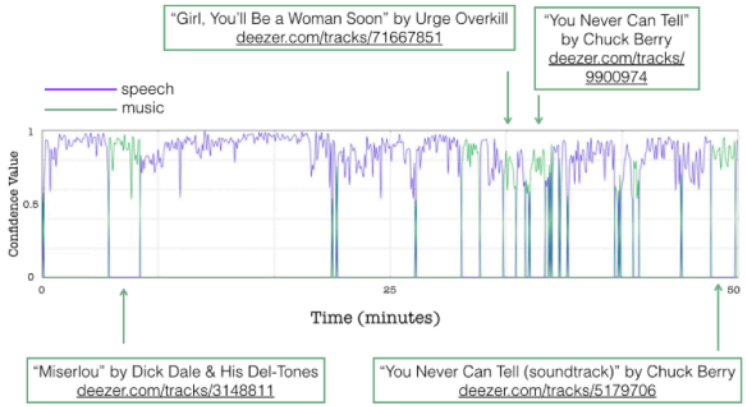


Figure 5.1: Example of song detection in the film "Pulp Fiction".

Source: ROYO-LETELIER Jimena "Detection and characterization of singing voice using deep neural networks", 2015, Report, Master-2 ATIAM, Deezer

Speech/music segmentation

Traditional approach

– **Goal**

- construct real-time computer system capable of distinguishing speech signals from music signals

– **Audio features**

- 13 different audio features
  - (1) 4 Hz modulation energy
  - (2) Percentage of "Low-energy" Frames
  - (3) Spectral Rolloff Point
  - (4) Spectral Centroid
  - (5) Spectral "Flux"
  - (6) Zero-Crossing Rate
  - (7) Cepstrum Resynthesis Residual Magnitude
  - (8) Pulse metric
  - + Variances over one-second window of (3) rolloff point, (4) spectral centroid, (5) spectral flux, (6) zero-crossing rate, and (7) cepstral resynthesis residual magnitude

CONSTRUCTION AND EVALUATION OF A ROBUST MULTIFEATURE SPEECH/MUSIC DISCRIMINATOR

Eric Scheirer[1] Malcolm Slaney[1]

Interval Research Corp., 1801-C Page Mill Road, Palo Alto, CA, 94304 USA

ABSTRACT

We report on the construction of a real-time computer system capable of distinguishing speech signals from music signals over a wide range of digital audio input. We have examined 13 features intended to measure conceptually distinct properties of speech and/or music signals, and combined them in several multidimensional classification frameworks. We provide extensive data on system performance and the cross-validated training/test setup used to evaluate the system. For the datasets currently in use, the best classifier classifies with 5.8% error on a frame-by-frame basis, and 1.4% error when integrating long (2.4 second) segments of sound.

1. OVERVIEW

The problem of distinguishing speech signals from music signals has become increasingly important as automatic speech recognition (ASR) systems are applied to more and more "real-world" multimedia domains. If we wish to build systems that perform ASR on soundtrack data, for example, it is important to be able to distinguish which segments of the soundtrack contain speech.

There has been some previous work on this topic [1], [2]. Some of this work has suggested features which prove valuable for incorporation into a multidimensional framework, and we have included them when possible. This paper extends that work in several ways: by considering multiple features, by examining powerful classification methods, and by describing a principled approach to training and testing the system.

The rest of our paper is divided into three sections: a description of the features examined in our system; a discussion of the different multivariate classification frameworks which we have evaluated, and results of a careful training and evaluation phase in which we present the performance characteristics of the system in its current state.

2. FEATURES

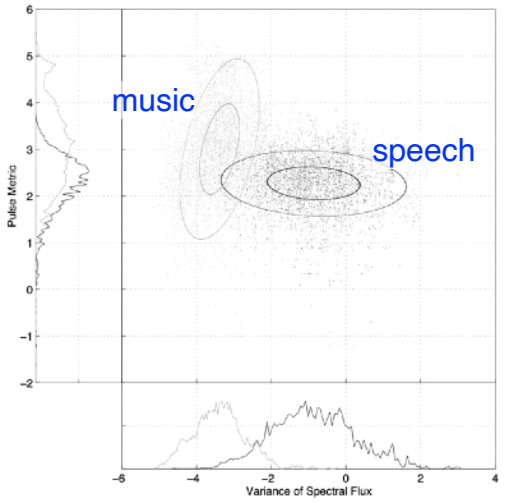
Thirteen features have been evaluated for use in the system. Each of them was intended to be a good discriminator on their own, as we shall show, not all of them end up adding value to a multivariate classifier. Of the thirteen, five are "variance" features, consisting of the variance in a one-second window of an underlying measure which is calculated on a single frame. If a feature has the property that it gives very different values for voiced and unvoiced speech, but remains relatively constant within a window of musical sound, then the variance of that feature will be a better discriminator than the feature itself.

It is also possible that other statistical analyses of "underlying" features, such as second or third central moments, skewness, kurtosis, and so forth, might make good features for discriminating classes of sound. For example, Saunders [2] bases four features on the zero-crossing rate, using the variance of the derivative, the third central moment, the thresholded value, and a skewness measure. The features used in this system are:

- 4 Hz modulation energy: Speech has a characteristic energy modulation peak around the 4 Hz syllabic rate [3]. We use a portion of the MFCC algorithm [4] to convert the audio signal into 40 perceptual channels. We extract the energy in each band, bandpass filter each channel with a second order filter with a center frequency of 4 Hz, then calculate the short-term energy by squaring and smoothing the result. We normalize each channel's 4 Hz energy by the overall channel energy in the frame, and sum the result from all channels. Speech tends to have more modulation energy at 4 Hz that music does.
- Percentage of "Low-Energy" Frames: The proportion of frames with RMS power less than 50% of the mean RMS power within a one-second window. The energy distribution for speech is more left-skewed than for music—there are more quiet frames—so this measure will be higher for speech than for music [2].
- Spectral Rolloff Point: The 95th percentile of the power spectral distribution. This measure distinguishes voiced from unvoiced speech—unvoiced speech has a high proportion of energy contained in the high-frequency range of the spectrum, where most of the energy for unvoiced speech and music is contained in lower bands. This is a measure of the "skewness" of the spectral shape—the value is higher for right-skewed distributions.
- Spectral Centroid: The "balancing point" of the spectral power distribution. Many kinds of music involve percussive sounds which, by including high-frequency noise, push the spectral mean higher. In addition, excitation energies can be higher for music than for speech, where pitch stays in a fairly low range. This measure gives different results for voiced and unvoiced speech.
- Spectral "Flux" (Delta Spectrum Magnitude): The 2-norm of the frame-to-frame spectral amplitude difference vector, $\|X_t - X_{t-1}\|$. Music has a higher rate of change, and goes through more drastic frame-to-frame changes than speech does; this value is higher for music than for speech. Note that speech alternates periods of transition (consonant-vowel boundaries) and periods of relative stasis (vowels), where music typically has a more constant rate of change. This method is somewhat similar to Hawley's, which attempts to detect harmonic continuity in music [1].
- Zero-Crossing Rate: The number of time-domain zero-crossings within a speech frame [2]. This is a correlate of the spectral centroid. Kedem [5] calls it a measure of the *dominant frequency* in a signal.
- Cepstrum Resynthesis Residual Magnitude: The 2-norm of the vector residual after cepstral analysis, smoothing, and resynthesis. Two do a real cepstral analysis [6] and smoothing of the spectrum, then resynthesize and compare the smoothed

[1]Eric Scheirer is currently at the MIT Media Laboratory, Cambridge, MA, USA. eds@media.mit.edu

[2]Malcolm Slaney can be reached at malcolm@interval.com

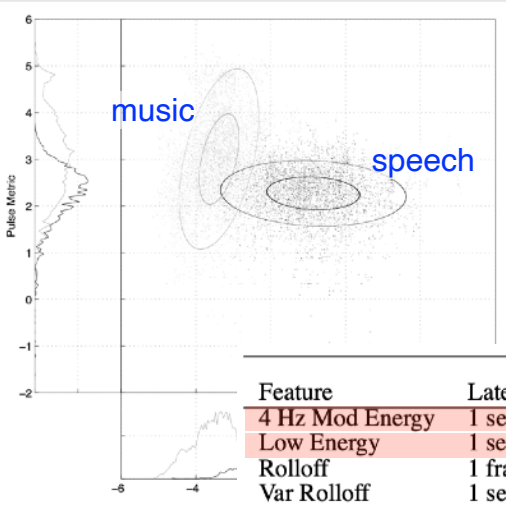


E. Scheirer and M. Slaney. Construction and evaluation of a robust multifeature speech/ music discriminator. In Proc. of IEEE ICASSP, 1997

Speech/music segmentation

Traditional approach (cont.)

- **ML algorithms**
  - Multi-dimensional Gaussian
  - Gaussian Mixture Model (GMM, diag Σ)
  - K-Nearest Neighbor
  - K-D spatial partitioning
- **Dataset**
  - 2 (speech/music) * 80 * 15-second- long audio samples from FM tuner
    - male, female, studio, telephonic
    - jazz, pop, country, salsa, reggae, classical, various non-Western styles, various sorts of rock, and new age music, both with and without vocals, in the music class.
- **Evaluation scenario**
  - N-fold cross-validation
    - 90% train, 10% test
- **Performance measures**
  - frame accuracy / frame error



Feature	Latency	CPU Time	Error
4 Hz Mod Energy	1 sec	18 %	12 ±1.7 %
Low Energy	1 sec	2 %	14 ±3.6 %
Rolloff	1 frame	17 %	46 ±2.9 %
Var Rolloff	1 sec	17 %	20 ±6.4 %
Spec Cent	1 frame	17 %	39 ±8.0 %
Var Spec Cent	1 sec	17 %	14 ±3.7 %
Spec Flux	1 frame	17 %	39 ±1.1 %
Var Spec Flux	1 sec	17 %	5.9 ±1.9 %
Zero-Cross Rate	1 frame	0 %	38 ±4.6 %
Var ZC Rate	1 sec	3 %	18 ±4.8 %
Ceps Resid	1 frame	46 %	37 ±7.5 %
Var Ceps Res	1 sec	47 %	22 ±5.7 %
Pulse Metric	5 sec	38 %	18 ±2.9 %

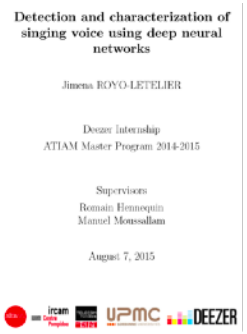
Framework	Speech Error	Music Error	Total Error
MAP G'ss'n	2.1 ±1.2 %	9.9 ±5.4 %	6.0 ±2.6 %
GMM: 1 G	3.0 ±1.1 %	8.7 ±6.6 %	5.8 ±2.9 %
5 G	3.2 ±1.1 %	8.4 ±6.8 %	5.8 ±2.9 %
20 G	3.4 ±1.5 %	7.7 ±6.2 %	5.6 ±2.4 %
50 G	3.0 ±1.4 %	8.2 ±6.6 %	5.6 ±2.6 %
kNN: $k = 1$	4.3 ±1.7 %	6.6 ±6.4 %	5.5 ±3.6 %
$k = 5$	4.2 ±1.8 %	6.4 ±6.2 %	5.3 ±3.5 %
$k = 11$	4.2 ±1.9 %	6.5 ±6.1 %	5.4 ±3.5 %
$k = 25$	4.2 ±1.9 %	6.5 ±6.1 %	5.4 ±3.5 %
k-d: $b = 5$	5.2 ±1.0 %	6.1 ±2.5 %	5.7 ±1.5 %
$b = 11$	5.4 ±1.1 %	6.1 ±2.8 %	5.8 ±1.6 %
$b = 21$	5.7 ±1.4 %	5.9 ±2.9 %	5.8 ±1.9 %
$b = 51$	5.9 ±1.8 %	5.5 ±2.7 %	5.7 ±2.0 %
$b = 101$	6.1 ±1.7 %	5.3 ±2.6 %	5.7 ±1.8 %

E. Scheirer and M. Slaney. Construction and evaluation of a robust multifeature speech/ music discriminator. In Proc. of IEEE ICASSP, 1997

Deep learning approach

- **Goal:**
  - replace hand-crafted features by deep learning

- **Test various audio inputs for DNN:**
  - CQT (dim=6 octaves * 24 bins= 144)
  - STFT (dim=700)
  - (M) MFFC (Mel Frequency Filter Coefficients) (dim=12)
  - (C) Chroma (dim=12)
  - (C) + (M) using the input depth
  - (C) + (M) + Δ(M) using the input depth

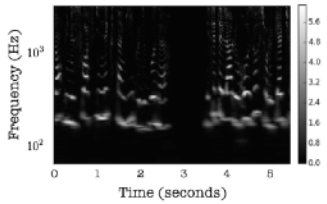

Detection and characterization of singing voice using deep neural networks

Jimena ROYO-LETELIER

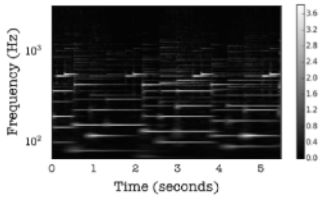
Deezer Internship
ATIAM Master Program 2014-2015

Supervisors
Romain Hennequin
Manuel Moussallam

August 7, 2015



(a) Speech sample "Lettre d'Alfred de Musset à George Sand: Pauvre George", deezer.com/track/67664893.



(b) Music sample "Desire, Musique Douce", deezer.com/track/57136771.

Figure 4.5: Typical CQT-grams of music and speech samples.

ROYO-LETELIER Jimena "Detection and characterization of singing voice using deep neural networks", 2015, Report, Master-2 ATIAM, Deezer

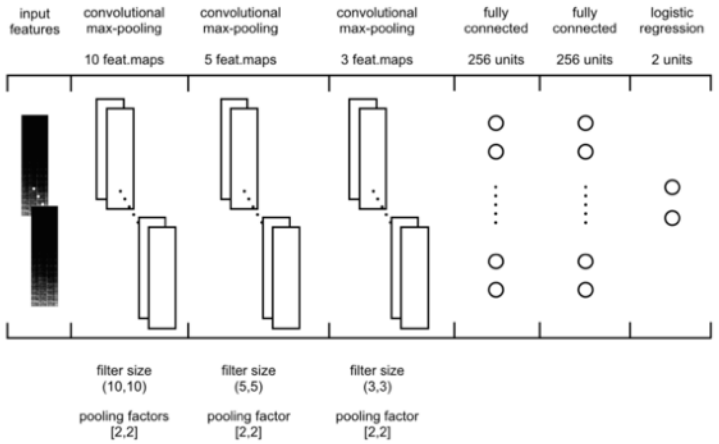
Speech/music segmentation

Deep learning approach (cont.)

- **ML algorithms:**
  - SVM
  - Random Forest (RF)
  - ConvNet (CNN)

- **Dataset:**
  - Large/Small: 41.960 * 5 seconds samples
  - Speech:
    - audiobooks, interviews, political speeches or pieces of theatre
  - music
    - Deezer catalogues, various genres

- **Performance measures**
  - Recall, Precision, F-measure



	Train	Validation	Test
Small dataset	8.000	472	472
Large dataset	40.000	980	980

	Input Shape	Input Mem. [GB]	Training [h]	Recall	Precision	F-measure
CQT	(1, 216, 144)	0.50	0.84	90.68	89.92	**90.30**
STFT	(1, 431, 700)	4.83	7.32	91.53	90.76	**91.14**
MFFC (M)	(1, 108, 12)	0.02	0.13	85.59	83.47	84.52
CHROMA (C)	(1, 108, 12)	0.02	0.12	75.86	88.00	81.48
(C)+(M)	(2, 108, 12)	0.04	0.13	93.10	84.38	88.52
(C)+(M)+Δ(M)	(3, 108, 12)	0.06	0.14	93.10	90.00	**91.53**

	Input Shape	Training [s]	Recall	Precision	F-measure
SVM	(1, 3888)	40	82.17	89.83	85.83
RF	(1, 3888)	16	76.27	89.11	82.19
CNN	(3, 108, 12)	**504**	93.10	90.00	**91.53**

ROYO-LETELIER Jimena "Detection and characterization of singing voice using deep neural networks", 2015, Report, Master-2 ATIAM, Deezer