Metric Learning for MIR

Part 1 - Foundations

Brian McFee Jongpil Lee Juhan Nam

NYU Center for Data Science

MARL

KAIST

Tutorial outline

- **Part 1: Foundations (Brian McFee)**
  - Linear metric learning methods
  - Overview of applications in MIR
  - Coding practice

- Part 2: Deep metric learning (Jongpil Lee)
  - Linear metric learning to deep metric learning
  - Advanced models
  - Coding practice

- Part 3: Variations and modern applications (Juhan Nam)
  - Cross-modality metric learning
  - Unsupervised learning

- Closing remarks

Similarity is everywhere inMIR!

- **Search** and **recommender systems** rely on similarity between songs, artists, albums, lyrics, tags, etc.
- **Cover song identification** uses similarity to determine if two recordings are of the same composition
- **Structural segmentation** uses similarity or dissimilarity between features to infer repetition, parallelism, and novelty
- **Nearest neighbor classifiers** use similarity to propagate information (eg, tags) from labeled to unlabeled data

Similarity is everywhere inMIR!

- **Search** and **recommender systems** rely on similarity between songs, artists, albums, etc.
- **Cover song** identification: measure similarity to determine if two songs are of the same comp
- **Structural** s res to infer repetition, parallelism, and novelty
- **Nearest neighbor classifiers** use similarity to propagate information (eg, tags) from labeled to unlabeled data

But what is “similarity”?

Similarity and metric learning

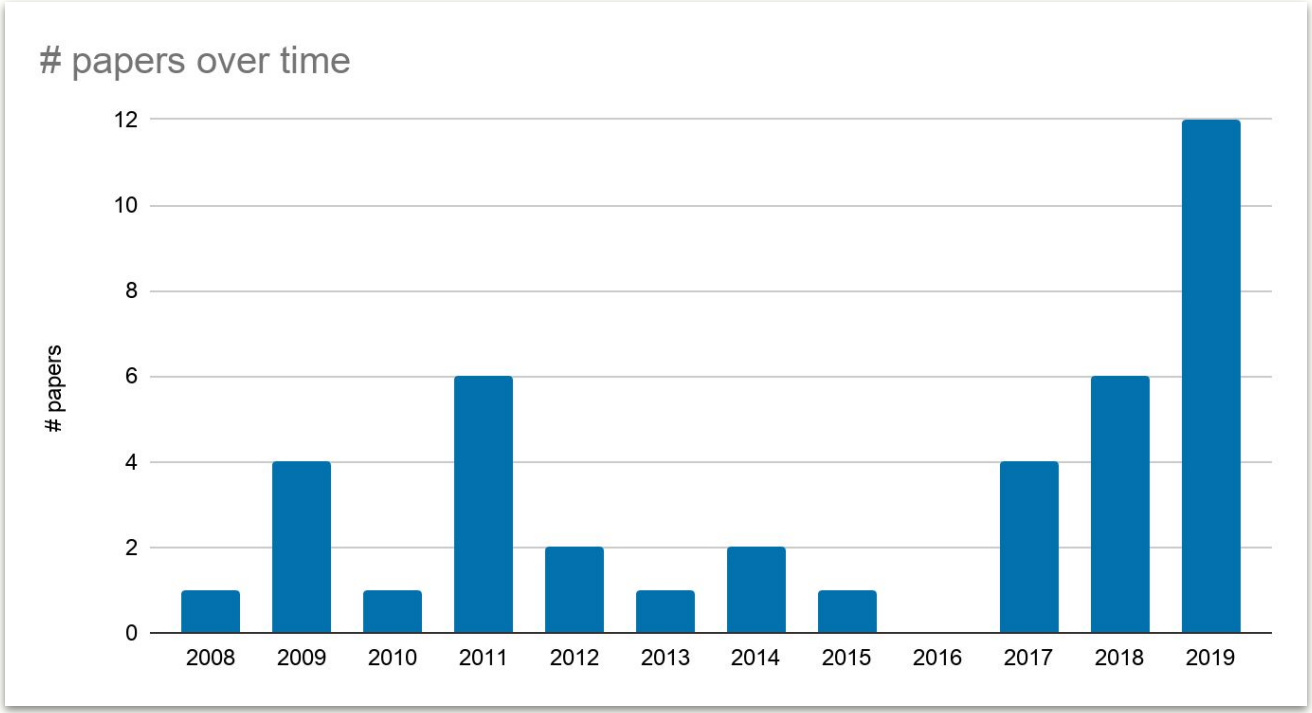
- For some applications, you can design an algorithm to compute similarity. But is it ***good***? Is it ***optimal***?
- **Metric learning** is the problem of learning how to compare items.
- We'll need some supervision (examples) to tell us what similarity should be. (a.k.a. Training data)
- From this, we'll learn a function to compare two items through a **distance metric**

Some reasons to use metric learning

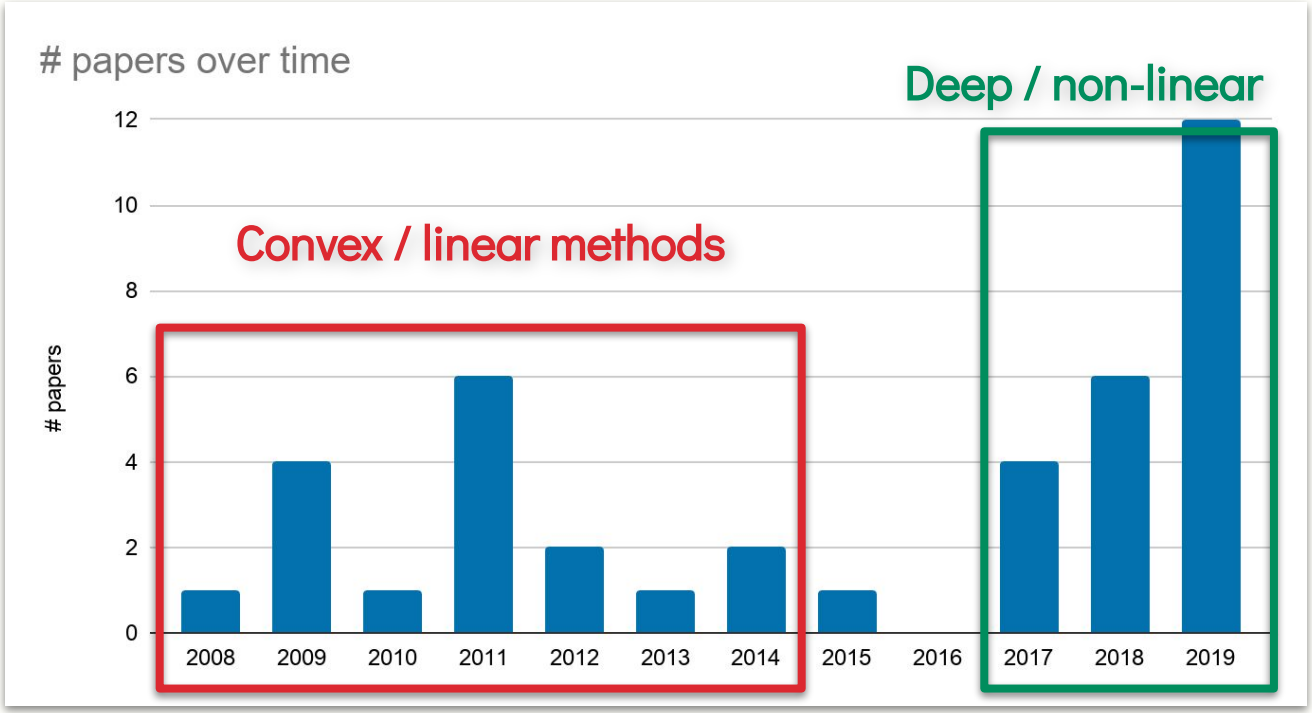
- Large or unbounded class vocabularies
  - Nearest neighbor methods will naturally support this
  - **Metric learning improves nearest neighbors!**
  - Example: cover detection

- Non-dichotomous notion of similarity
  - **Metric learning can be trained for rank-ordering data, not just same/different!**
  - Example: recommendation and similarity judgments

- Your classification problem naturally involves pairs of items
  - **Distance between vector representations is a flexible model**
  - Example: structure analysis



Timeline of metric learning in MIR (ISMIR, ICASSP, & related venues)



Timeline of metric learning in MIR (ISMIR, ICASSP, & related venues)

Distance metrics

- Let $x_1, x_2 \in \mathbb{R}^D$ denote two data points to be compared
- The (squared) Euclidean distance is

$$\|x_1 - x_2\|^2 = (x_1 - x_2)^\top (x_1 - x_2) = \sum_d (x_1[d] - x_2[d])^2$$

Distance metrics

- Let $x_1, x_2 \in \mathbb{R}^D$ denote two data points to be compared
- The (squared) Euclidean distance is

$$\|x_1 - x_2\|^2 = (x_1 - x_2)^\top (x_1 - x_2) = \sum_d (x_1[d] - x_2[d])^2$$

- **Linear metric learning** seeks a projection $M \in \mathbb{R}^{D \times D}$:

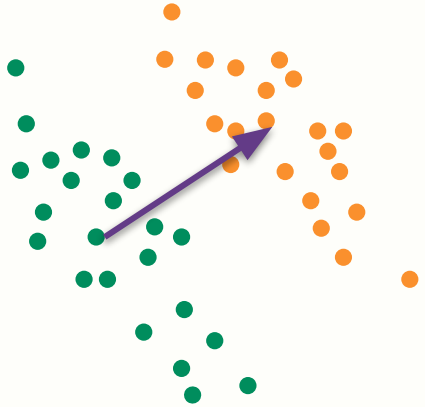
$$\|Mx_1 - Mx_2\|^2 = (Mx_1 - Mx_2)^\top (Mx_1 - Mx_2) = (x_1 - x_2)^\top M^\top M (x_1 - x_2)$$

- After projection, **similar** points should have **small distance** and **dissimilar** points should have **large distance**

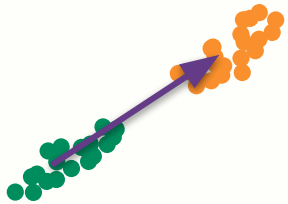
Linear discriminant analysis

[Fisher; 1935]

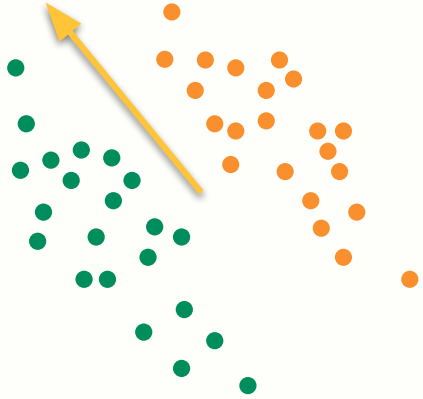
- Assume we have data in two classes: $Y = \{0, 1\}$



- **Minimize** distance between points from the **same class**
- **Maximize** distance between points of **different classes**
- If two classes are **linearly separable**, then **one direction** is enough.
  Using more dimensions would only increase distances.



LDA

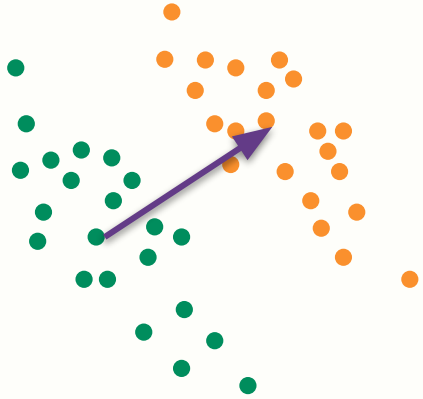


- "Average (squared) distance" is **variance!**
- Find a direction that simultaneously minimizes **within-class** variance and maximizes **between-class** variance

- Principal components analysis (PCA): direction(s) to **maximize total variance**
  ⇒ leading eigenvector(s) of the covariance matrix Σ

$$\max_{m} m^{\top}\Sigma m \quad \text{such that} \quad \|m\| = 1$$

LDA



- "Average (squared) distance" is **variance!**
- Find a direction that simultaneously minimizes **within-class** variance and maximizes **between-class** variance
- $\min_m m^T(\Sigma_0 + \Sigma_1)m \;/\; m^T(\Sigma_{01})m$
  s.t. $\|m\| = 1$
- This can be solved efficiently by "generalized Eigenvectors": $(\Sigma_0 + \Sigma_1)m = \lambda\Sigma_{01}m$

Relaxing the assumptions

- LDA makes **strong assumptions** about the data
  - Gaussianity + Linear separability
- **Metric learning** is well suited to **nearest neighbor** methods, which don't need those assumptions!
- Starting with [Xing et al., 2002], there are now many algorithms to learn distance metrics in various settings
- Algorithms differ mainly in what form of **supervision** is required, and how exactly distances are compared / optimized

Different kinds of supervision

- Class labels:

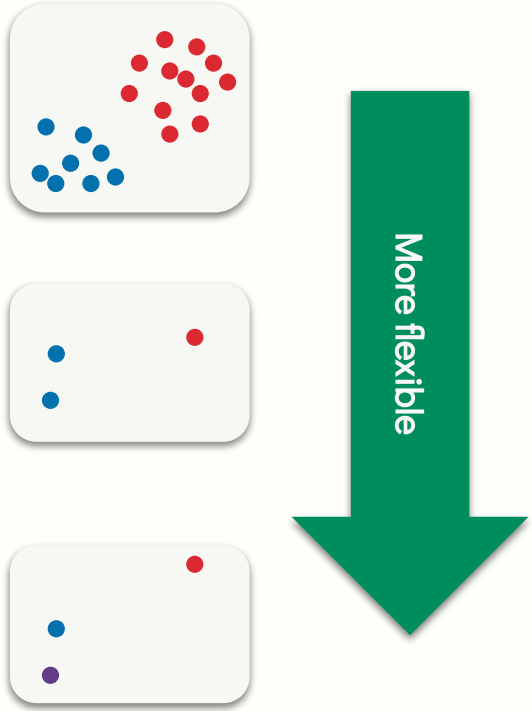
  (x, y)

- Pairwise **similarity**/**dissimilarity**:

  $(x_1, x_2, \pm)$

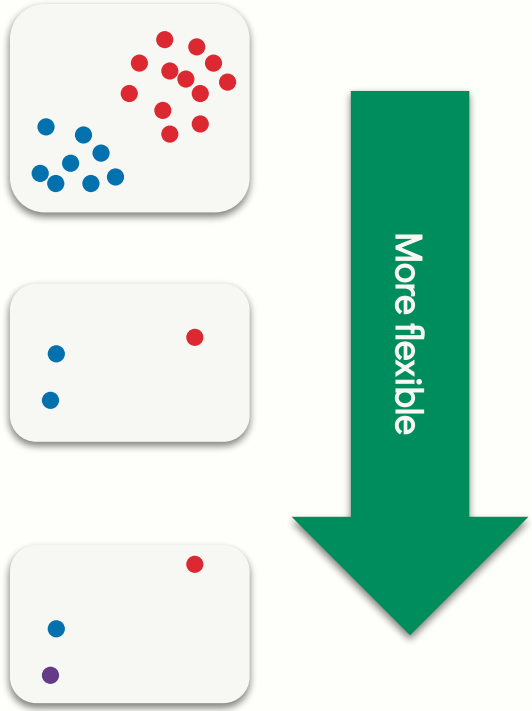
- **Relative** comparisons (**triplets**):

  $(x_1, x_2, x_3) \quad \Rightarrow \quad d(x_1, x_2) < d(x_1, x_3)$



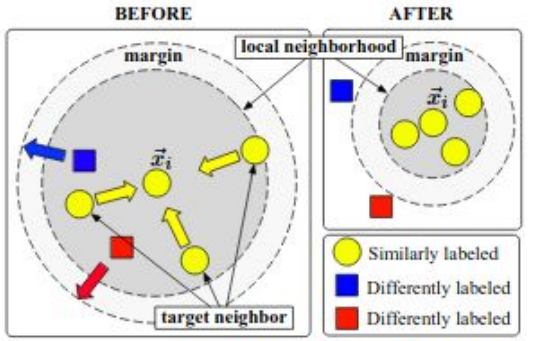
What supervision is right for you?

- Is it reasonable for **all examples of a class** to cluster together?
- Or are locally consistent neighborhoods good enough? Can you identify good pairs?
- Do you even have class labels, or some other source of similarity judgments?



Large Margin Nearest Neighbors*

[Weinberger, Blitzer, Saul; 2006]



- For each example x_i, find k nearest "target" neighbors $\{x_j\}$ with the **same label** as x_i

- Each of these targets should be closer to x_i than any other **differently labeled point x_k by at least a margin**

$$\min_M \sum_{(i,j,k)} \max(0,\ \|Mx_i - Mx_j\|^2 - \|Mx_i - Mx_k\|^2 + 1)$$

- Optimize M by (stochastic) gradient descent on training data

*simplified for presentation

Coding time!

Summary of part 1

- We've seen how to learn a **linear transformation** of feature vectors that improves nearest neighbor classification / retrieval
- By learning the distance metric for a given task, we can be more inclusive of different features and representations.
- In part 2, we'll see how the same basic concepts can be extended to **non-linear transformations** using deep learning!

Further readings

- Bellet, Aurélien, Amaury Habrard, and Marc Sebban. "A survey on metric learning for feature vectors and structured data." arXiv preprint arXiv:1306.6709 (2013).

- Kulis, Brian. "Metric learning: A survey." Foundations and trends in machine learning 5.4 (2012): 287-364.

Tutorial outline

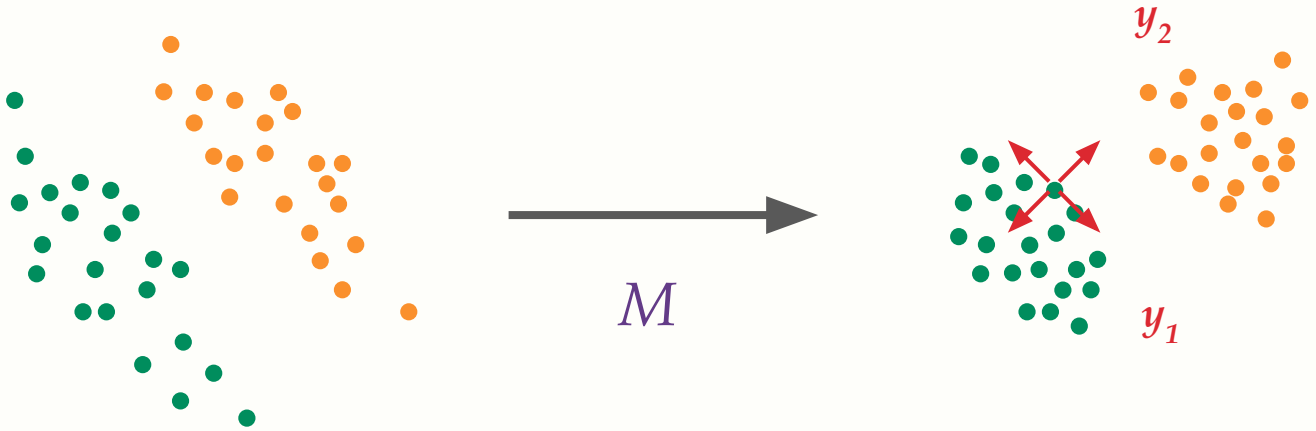
- Part 1: Foundations (Brian McFee)
  - Linear metric learning methods
  - Overview of applications in MIR
  - Coding practice

- **Part 2: Deep metric learning (Jongpil Lee)**
  - Linear metric learning to deep metric learning
  - Advanced models
  - Coding practice

- Part 3: Variations and modern applications (Juhan Nam)
  - Cross-modality metric learning
  - Unsupervised learning

- Closing remarks

Linear metric learning



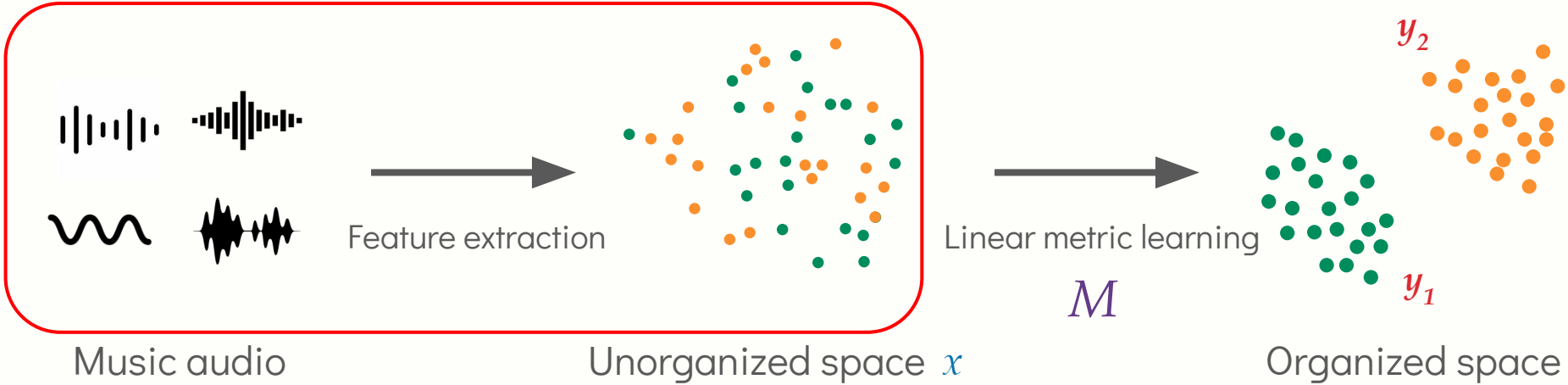
Large Margin Nearest Neighbors (LMNN)

$$(x, y) \text{ or } (x_i, x_j, x_k) \Rightarrow d(x_i, x_j) < d(x_i, x_k)$$

$$\|Mx_1 - Mx_2\|^2 = (Mx_1 - Mx_2)^\top (Mx_1 - Mx_2) = (x_1 - x_2)^\top M^\top M (x_1 - x_2)$$

$$\min_M \sum_{(i,j,k)} \max\left(0, \|Mx_i - Mx_j\|^2 - \|Mx_i - Mx_k\|^2 + 1\right)$$

Linear metric learning to deep metric learning

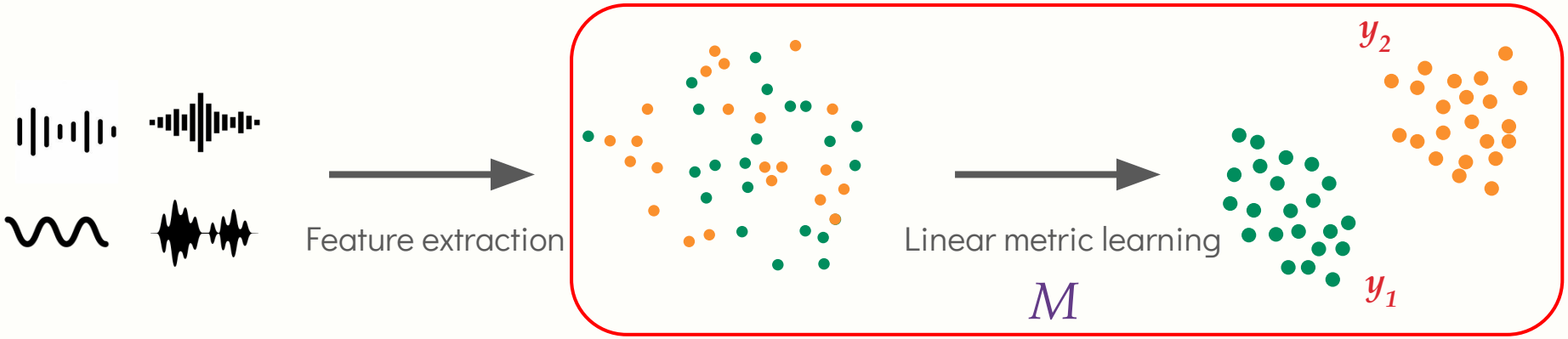


$$(x, y) \text{ or } (x_i, x_j, x_k) \Rightarrow d(x_i, x_j) < d(x_i, x_k)$$

$$\|Mx_1 - Mx_2\|^2 = (Mx_1 - Mx_2)^\top (Mx_1 - Mx_2) = (x_1 - x_2)^\top M^\top M (x_1 - x_2)$$

$$\min_M \sum_{(i,j,k)} \max\left(0, \|Mx_i - Mx_j\|^2 - \|Mx_i - Mx_k\|^2 + 1\right)$$

Linear metric learning to deep metric learning

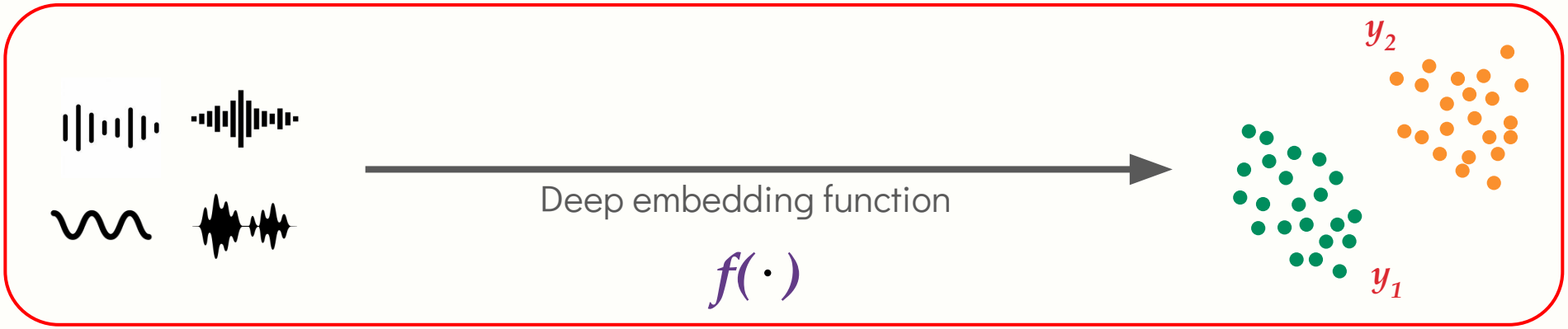


(x, y) or $(x_i, x_j, x_k) \Rightarrow d(x_i, x_j) < d(x_i, x_k)$

$$\|Mx_1 - Mx_2\|^2 = (Mx_1 - Mx_2)^\top (Mx_1 - Mx_2) = (x_1 - x_2)^\top M^\top M (x_1 - x_2)$$

$$\min_M \sum_{(i,j,k)} \max\left(0, \|Mx_i - Mx_j\|^2 - \|Mx_i - Mx_k\|^2 + 1\right)$$

Linear metric learning to deep metric learning



Music audio x

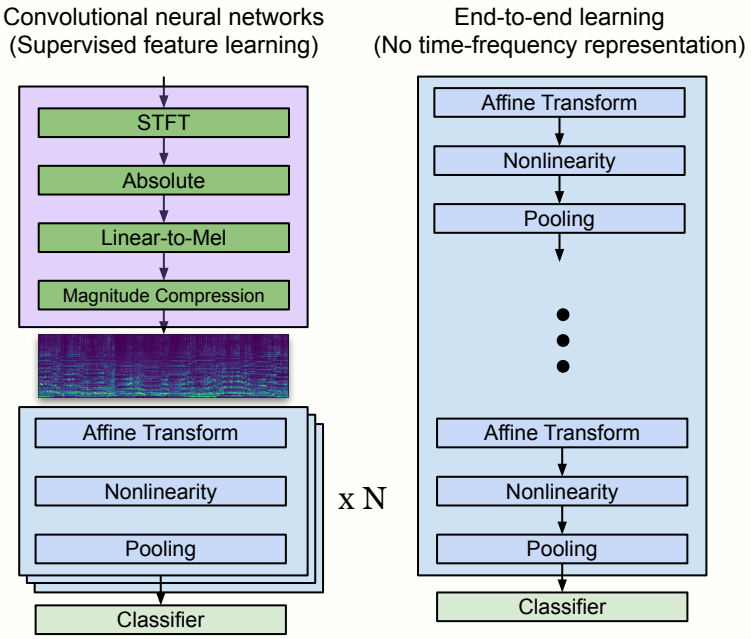
$$(x, y) \text{ or } (x_i, x_j, x_k) \Rightarrow d(x_i, x_j) < d(x_i, x_k)$$

$$D(f(x_1), f(x_2)) = \|f(x_1) - f(x_2)\|^2, \textit{ distance metric or similarity metric}$$

$$\min_f \sum_{(i,j,k)} \max(0, \|f(x_i) - f(x_j)\|^2 - \|f(x_i) - f(x_k)\|^2 + \alpha)$$

Deep embedding function $f(\cdot)$ (backbone model)

[Humphrey et al., 2013, Nam et al., 2018, Pons et al., 2019]

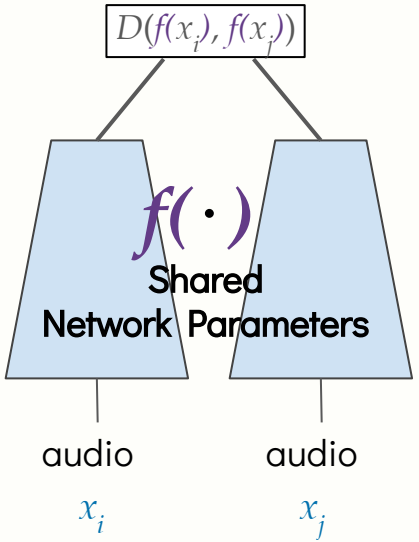


- Any deep nonlinear embedding function
- Gradient descent optimization
- Audio, deep embedding features

Siamese networks

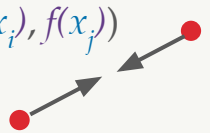
[Hadsell et al., 2006]

$$D(f(x_i), f(x_j)) = \|f(x_i) - f(x_j)\|_2$$

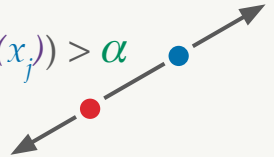


$(x_i, x_j, y) \Rightarrow$

Minimize $D(f(x_i), f(x_j))$, if $y = 1$



$D(f(x_i), f(x_j)) > \alpha$, if $y = 0$

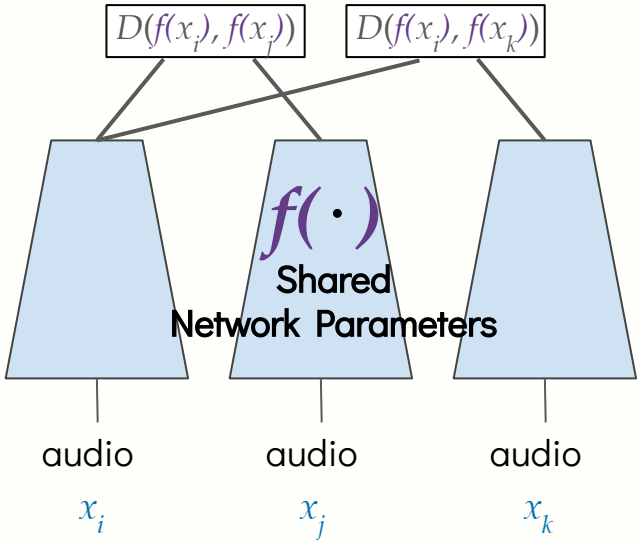


Contrastive loss

$$\min_f \sum_{(i,j,y)} (1 - y)\frac{1}{2} D(f(x_i), f(x_j))^2 + (y)\ \frac{1}{2}\max\left(0, \alpha - D(f(x_i), f(x_j))\right)^2$$

Triplet networks

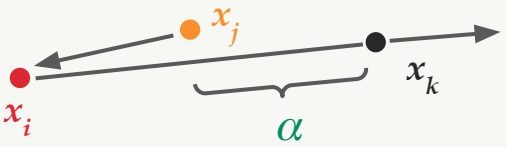
[Hoffer and Ailon, 2015]



$$D(f(x_i), f(x_j)) = \|f(x_i) - f(x_j)\|^2$$

x_i, x_j are the same class, not x_k

$$(x_i, x_j, x_k) \Rightarrow D(f(x_i), f(x_j)) + \alpha < D(f(x_i), f(x_k))$$



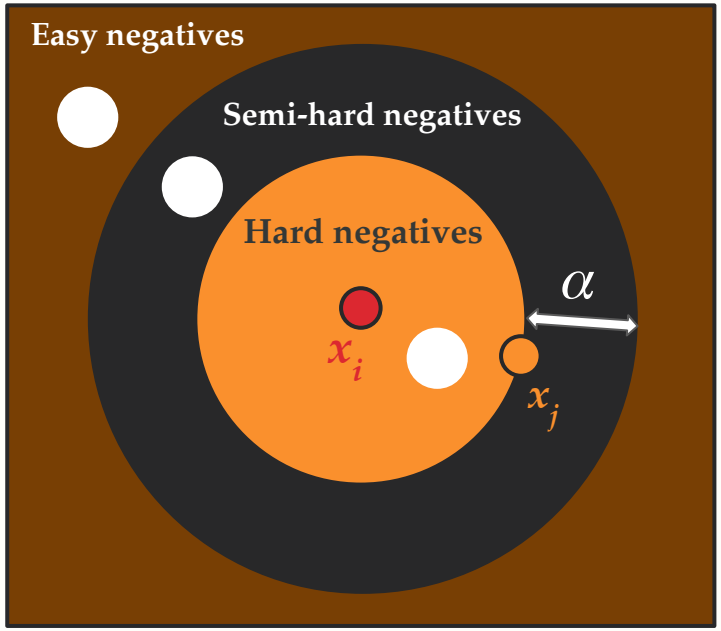
Triplet hinge loss

$$\min_f \sum_{(i,j,k)} \max\left(0,\ D(f(x_i), f(x_j)) - D(f(x_i), f(x_k)) + \alpha\right)$$

Triplet sampling matters

[Wu et al., 2017]

(x_i, x_j, x_k) $\begin{cases} x_i: \text{ anchor sample} \\ x_j: \text{ positive sample} \\ x_k: \text{ negative sample} \end{cases}$



- Easy negatives

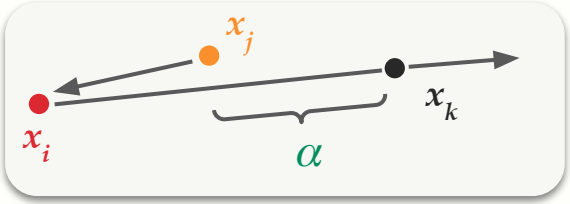
$$D(f(x_i), f(x_j)) + \alpha < D(f(x_i), f(x_k))$$

- Hard negatives

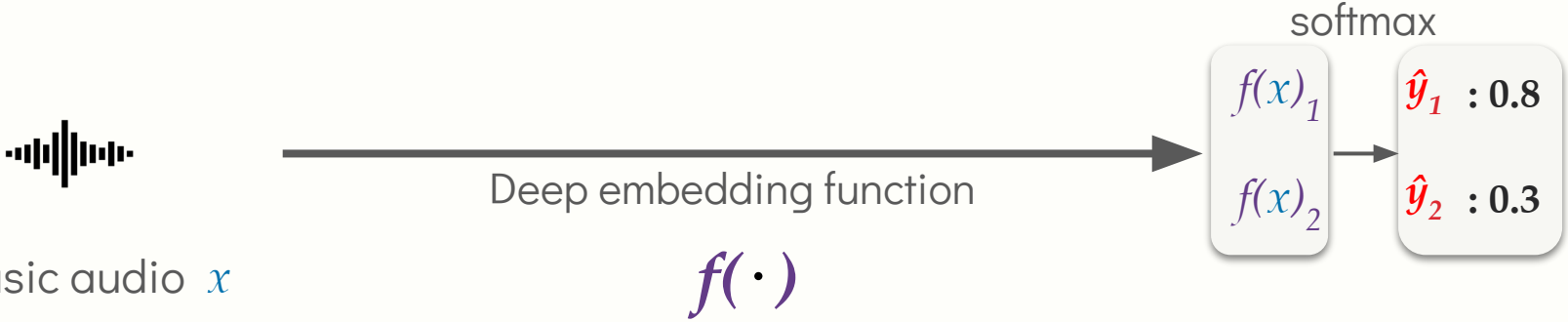
$$D(f(x_i), f(x_j)) > D(f(x_i), f(x_k))$$

- Semi-hard negatives

$$D(f(x_i), f(x_j)) < D(f(x_i), f(x_k)) < D(f(x_i), f(x_j)) + \alpha$$



Classification



$$(x, y)$$

$$\hat{y}_z = \text{softmax}(f(x)_z) = \frac{\exp(f(x)_z)}{\sum_z \exp(f(x)_z)}$$

$$\min_f - \sum_z y_z \log(\hat{y}_z)$$

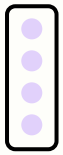
Classification

Music audio x

Deep embedding function

$f_{n-1}(\cdot)$

Last hidden layer features



$f_{n-1}(x)$

The last hidden layer

$h = [c_1, c_2]$

softmax

$f(x)_1$

$f(x)_2$

$\hat{y}_1$: 0.8

$\hat{y}_2$: 0.3

where c_z is a centroid of class z

$$f(x) = f_{n-1}(x) \square \quad h$$

$$f(x)_z = f_{n-1}(x) \square \quad c_z$$

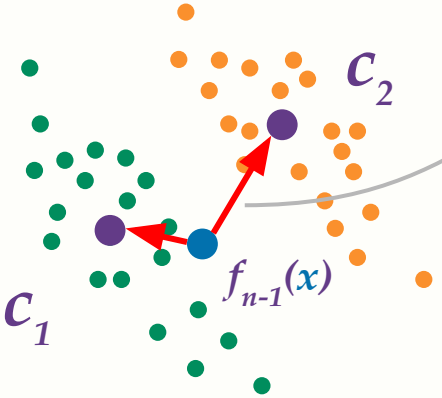
$$\hat{y}_z = \text{softmax}(f_{n-1}(x) \square c_z) = \frac{\exp(f_{n-1}(x) \square c_z)}{\sum_z \exp(f_{n-1}(x) \square c_z)}$$

$$\min_f - \sum_z y_z \log(\hat{y}_z)$$

*simplified for presentation

Classification

[Movshovitz-Attias et al., 2017, Zhai and Wu, 2018, Lee et al., 2020]



$$D(f_{n-1}(x), C_z) = f_{n-1}(x) \square C_z$$

Metric space $f_{n-1}(x)$ is not in normalized space

=> poor nearest neighbor search performance

$$D(f_{n-1}(x), C_z) = f_{n-1}(x) \square C_z$$

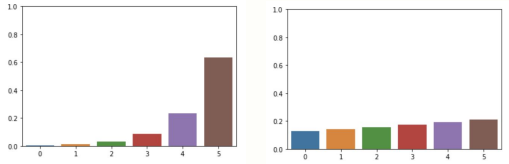
$$\hat{y}_z = \text{softmax}(D(f_{n-1}(x), C_z))$$

$$\min_f - \sum_z y_z \log(\hat{y}_z)$$

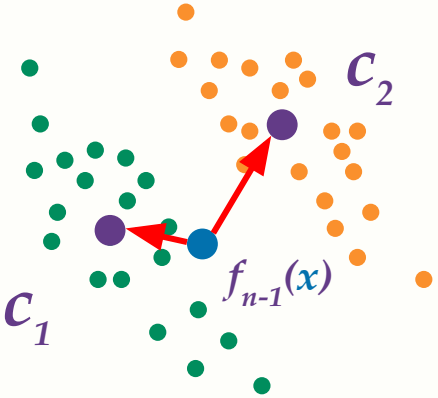
Classification in metric learning

[Movshovitz-Attias et al., 2017, Zhai and Wu, 2018, Lee et al., 2020]

Output probability $\hat{y}$



when D is in appropriate range when D is small



$$D(f_{n-1}(x), c_z) = \frac{f_{n-1}(x)}{\|f_{n-1}(x)\|} \cdot \frac{c_z}{\|c_z\|} \cdot \frac{1}{\tau}$$

or

$$D(f_{n-1}(x), c_z) = f_{n-1}(x) \cdot c_z$$

$$\hat{y}_z = \text{softmax}(D(f_{n-1}(x), c_z))$$

$$\min_f - \sum_z y_z \log(\hat{y}_z)$$

→

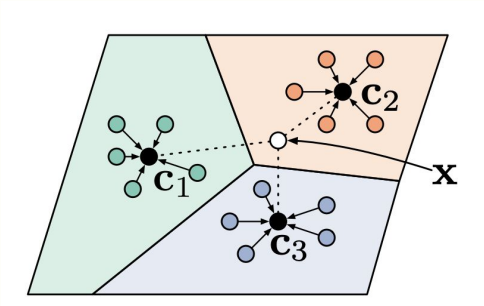
$$D(f_{n-1}(x), c_z) = \frac{f_{n-1}(x)}{\|f_{n-1}(x)\|} \cdot c_z$$

Add a normalization layer to the last hidden layer

Prototypical networks

[Snell et al., 2017]

- Prototype of a class is the mean vector of the embedding features belonging to that class
- Few-shot classification



Non-parametric prototype

$$c_z = \frac{1}{N} \sum_{i \in c_z}^{N} f(x_i)$$

$$D(f(x), c_z) = \|f(x) - c_z\|^2$$

$$\hat{y}_z = \text{softmax}(-D(f(x), c_z)) = \frac{\exp(-D(f(x), c_z))}{\sum_z \exp(-D(f(x), c_z))}$$

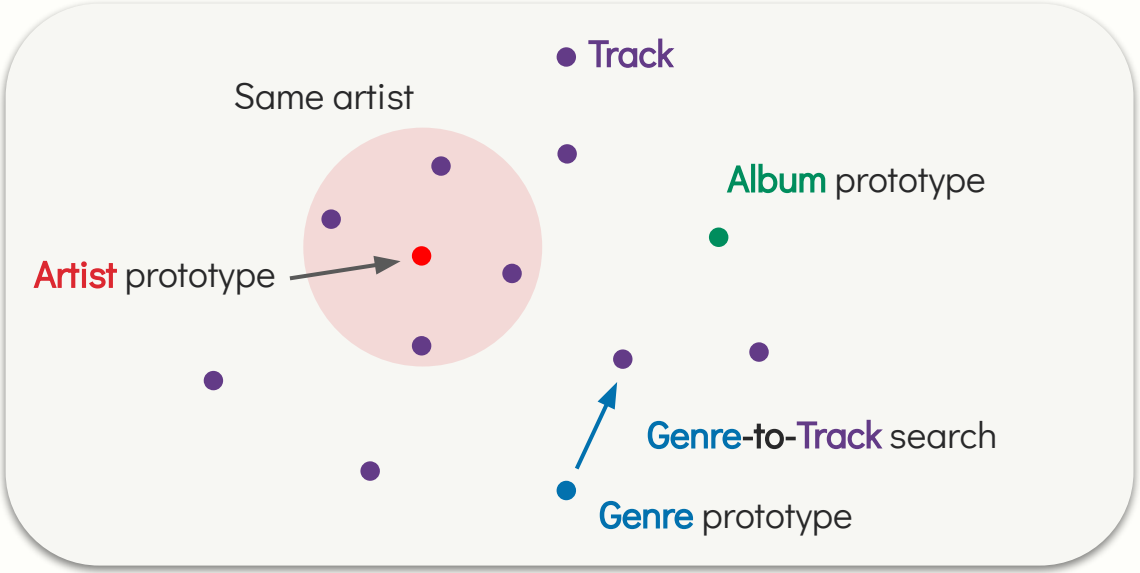
*simplified for presentation

Use of prototypes

- If we have a trained embedding function $f(\cdot)$, we can get the prototypes of the category by averaging all the embedding features belonging to the same category and use them for search

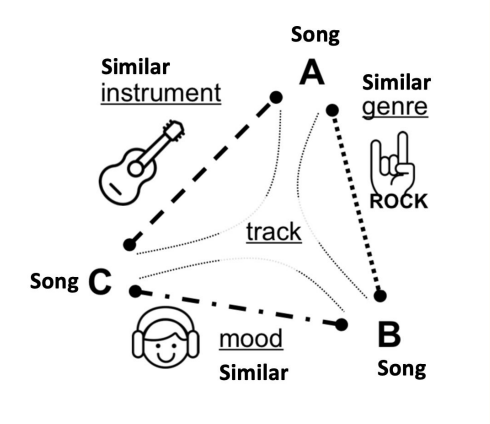
- Music items
  - Track
  - Album
  - Artist
  - Tag

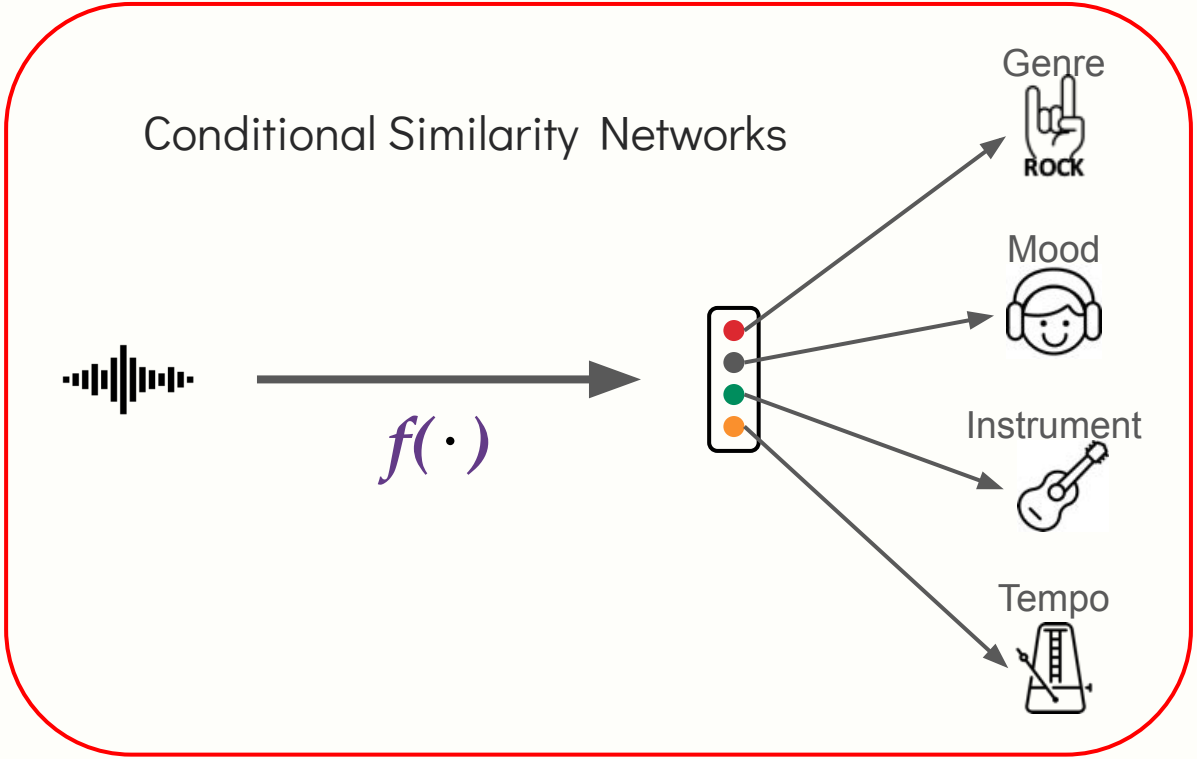
- Cross item search
  - Tag-to-Track
  - Artist-to-Track
  - Album-to-Track



Disentangled metric learning

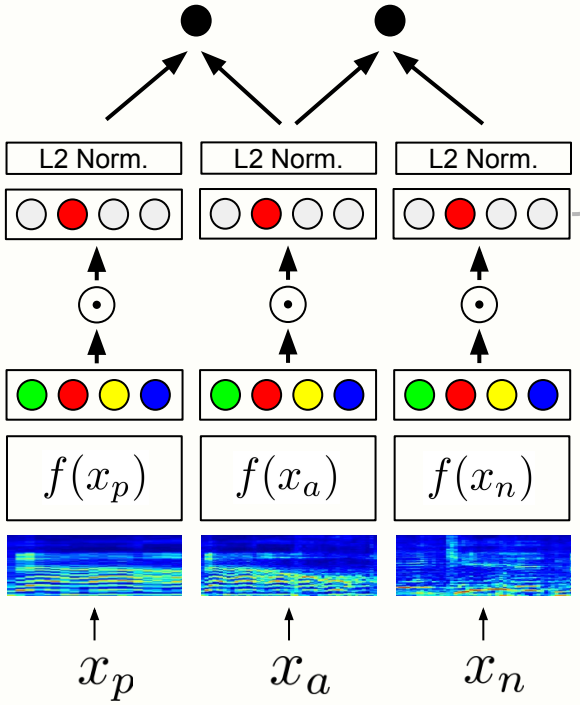
[Veit et al., 2017, Lee et al., 2020]





Disentangled metric learning

[Veit et al., 2017, Lee et al., 2020]



Mask the embedding features

depending on the notion of similarity s

(e.g. genre, mood, instruments, tempo)

$$D(f(x_i), f(x_j)) = \|f(x_i) - f(x_j)\|^2$$

$$(x_a, x_p, x_n; s)$$

$$D(f(x_i), f(x_j); s) = \|f(x_i) \circ m_s - f(x_j) \circ m_s\|^2$$

Unsupervised metric learning

[Dosovitskiy et al., 2014, Wu et al., 2018]

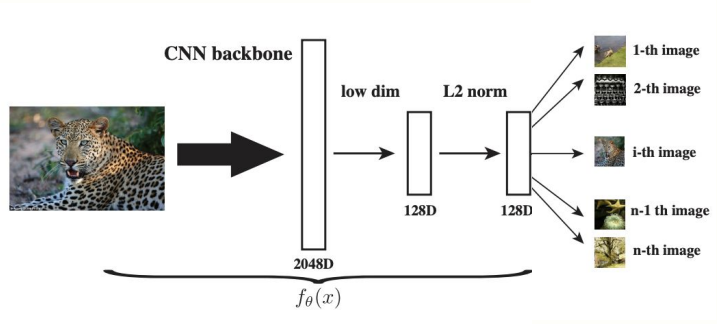
What if there are too many training instances? (e.g. millions of training instances)

- Instead of training the model using category labels, you can train the model to discriminate between instances
- Classification: augmented instance $x_i \rightarrow \tilde{x}_i$ and centroid of instances C_i

$$D(f(\tilde{x}_i), C_i) = f(\tilde{x}_i) \square C_i$$

$$\hat{y}_i = \text{softmax}(D(f(\tilde{x}_i), C_i)) = \frac{\exp(D(f(\tilde{x}_i), C_i))}{\sum_i \exp(D(f(\tilde{x}_i), C_i))}$$

$$\min_f - \sum_i y_i \log(\hat{y}_i)$$



*simplified for presentation

Unsupervised metric learning

[Chen et al., 2020]

- Instead of using centroid of instances, you can directly compare instances within batch sampling

- Metric learning: Augmented instance $x_i \rightarrow \tilde{x}_i$ and augmented instance $x_j \rightarrow \tilde{x}_j$

$$D(f(\tilde{x}_i), f(\tilde{x}_j)) = cos(f(\tilde{x}_i), f(\tilde{x}_j))$$

$$\hat{y}_{i,j} = \frac{\exp(D(f(\tilde{x}_i), f(\tilde{x}_j)) \ / \tau)}{\sum_k 1_{[k \neq i]} \exp(D(f(\tilde{x}_i), f(\tilde{x}_k)) \ / \tau)}$$

$$\min_f - \sum_{i,j} y_{i,j} \log(\hat{y}_{i,j})$$

- Track
- Album
- Artist

*simplified for presentation

Applications in MIR

- Similarity pairs or triplets [Lu et al., 2017]
  - (x_i, x_j, x_k)
- Semantic labels (tag, genre) [Schindler and Knees, 2019, Lee et al., 2020]
  - **Mostly Multi-labeled**
  - $(x, y) => (x_i, x_j, x_k)$

    Latent space similarity or whether to share the same tag
- Objective metadata (artist, album, track) [Park et al., 2018, Royo-Letelier et al., 2018, Lee et al., 2019]
  - **Free annotations**
  - **The number of classes is very large**

Semantic labels

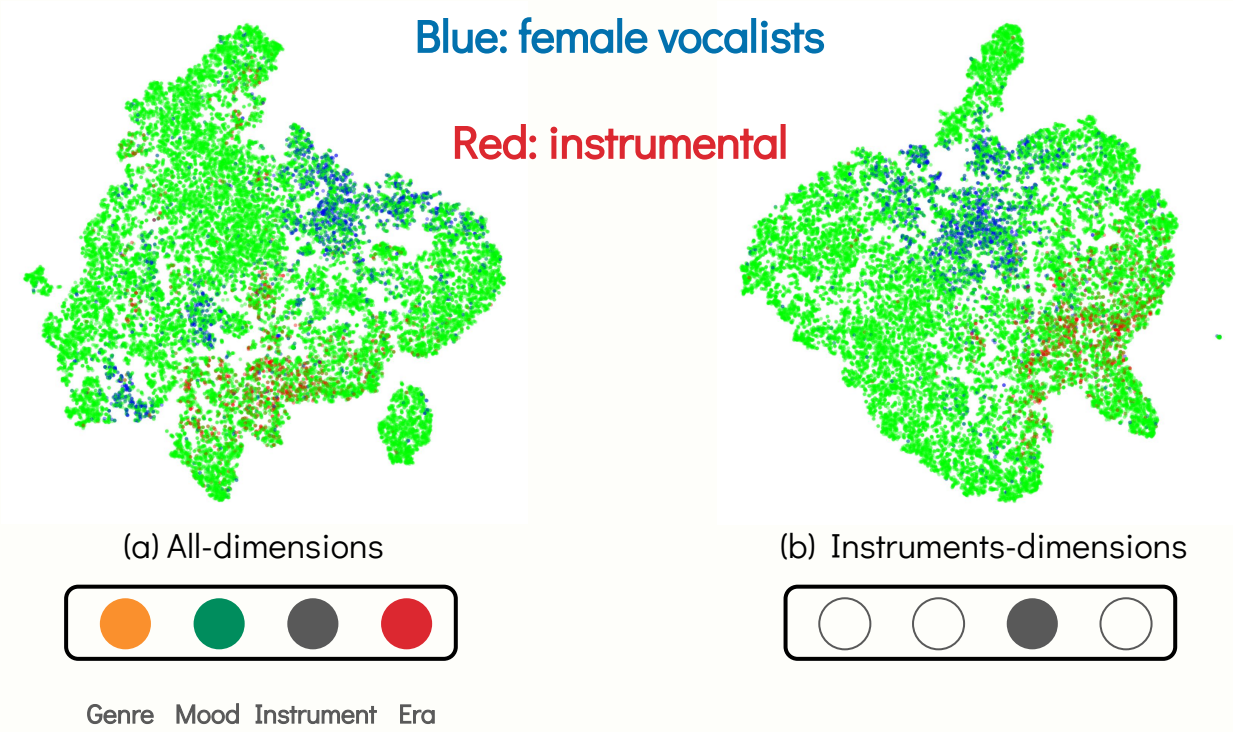
[Lee et al., 2020]

	Normalization	Similarity-based retrieval				Auto-tagging AUC
		R@1	R@2	R@4	R@8	
Triplet	yes	31.8	45.2	59.9	73.0	0.815
Classification	no	6.1	11.5	21.1	35.9	0.887
Classification	yes	43.8	57.8	70.3	80.3	0.887

- Triplet networks versus classification model using tag labels
- Evaluated in both classification (auto-tagging) and similarity-based retrieval
- For tagging evaluation of triplet networks, they measure the distance between the example and the class centroids and use them as class predictions
- The results show that
  - Triplet networks perform better in similarity-based retrieval than classification model without a normalization layer
  - However, classification model shows better performance on auto-tagging
  - **Adding a normalization layer makes the classification model performs well on both tasks**

Disentangled metric learning

[Lee et al., 2020]



Objective metadata

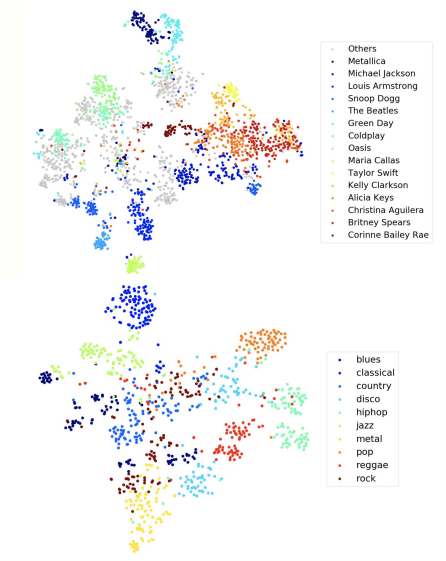
[Park et al., 2018]

	Tag-label (classification)	Artist-label (classification)	Artist-label (triplet)
GTZAN (fault-filtered)	0.6759	0.6655	0.6966
FMA small	0.5332	0.5269	0.5732

- Triplet networks versus classification model using artist labels
- Also, they are compared with classification model using tag labels
- The models are evaluated with a k-nearest neighbor genre classification based on embedding feature similarity to the held-out dataset
- The results show that
  - Artist-label classification model almost matches the performance of tag-label classification model
  - **Artist label triplet model outperforms the two classification models**

Clustering and prototype search

[Park et al., 2018]



Artist prototype nearest neighbor search

query		HIP-HOP		Rock Band		Pop rock
	0 (1.0) Eminem		0 (1.0) Green Day		0 (1.0) John Lennon	
	1 (0.9182) OutKast		1 (0.8728) Unwritten Law		1 (0.8355) Cliff Richard	
	2 (0.9027) Spezializtz		2 (0.8692) P.O.D.		2 (0.8199) The Who	
	3 (0.9012) Dino MC 47		3 (0.8237) Tokyo Rose		3 (0.7959) Status Quo	
	4 (0.8878) Bone Thugs-N-Harmony		4 (0.7903) Linkin Park		4 (0.7836) Nick Cave and the Bad Seeds	
	5 (0.8840) Method Man		5 (0.7532) All Star United		5 (0.7772) T.Love	
	6 (0.8814) Big Punisher		6 (0.7532) Hinder		6 (0.7713) Blake Morgan	
	7 (0.8761) Cor Veleno		7 (0.7420) The All-American Rejects		7 (0.7538) Radney Foster	
	8 (0.8642) Lil' Wayne		8 (0.7384) Jimmy Eat World		8 (0.7509) Coldplay	
	9 (0.845) Obie Trice		9 (0.7206) Third Day		9 (0.7409) The Beach Boys	
	10 (0.8301) mcenroe & Birdapres		10 (0.7000) Bon Jovi		10 (0.7391) Badly Drawn Boy	

Coding time!

Summary of part 2

- We've extended linear metric learning to **deep metric learning**
- We've also introduced some advanced models and basic applications in MIR.
- In part 3, we'll see how the deep metric learning concepts can be extended to use **diverse music modalities** (e.g. audio, image, MIDI or text)!

Further readings

- Kaya, Mahmut, and Bílge, Hasan Şakir. "Deep metric learning: A survey." Symmetry 11.9 (2019): 1066.

- Grosche, Peter, Müller, Meinard, and Serrà, Joan. "Audio content-based music retrieval." Dagstuhl Follow-Ups. Vol.3, Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2012).

Tutorial outline

- Part 1: Foundations (Brian McFee)
  - Linear metric learning methods
  - Overview of applications in MIR
  - Coding practice

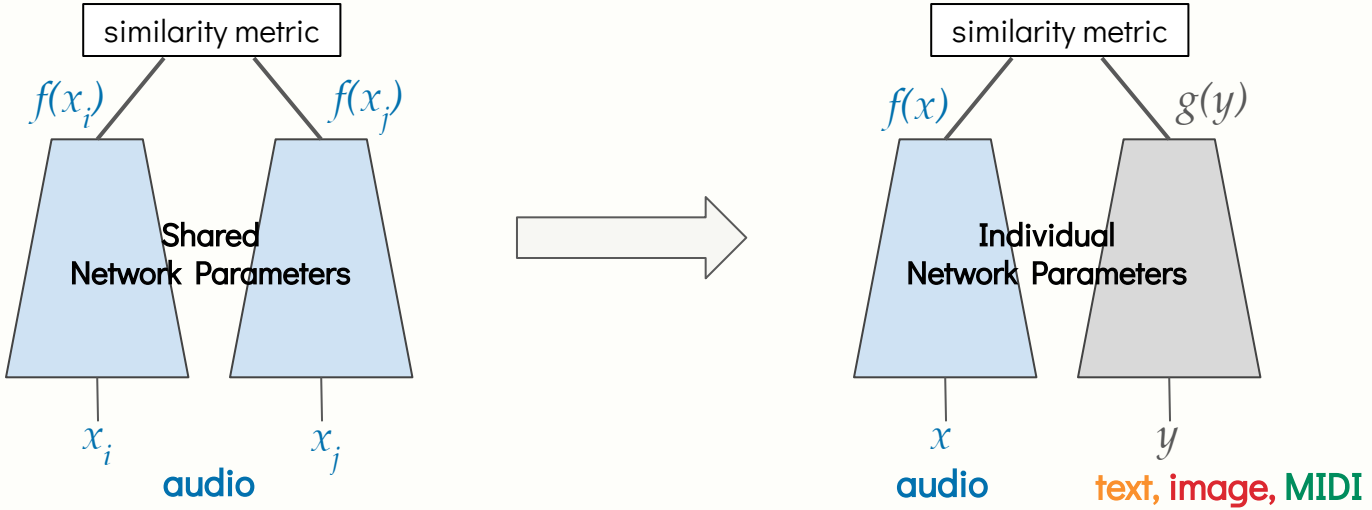
- Part 2: Deep metric learning (Jongpil Lee)
  - Linear metric learning to deep metric learning
  - Advanced models
  - Coding practice

- **Part 3: Variations and modern applications (Juhan Nam)**
  - Cross-modality metric learning
  - Unsupervised learning

- Closing remarks

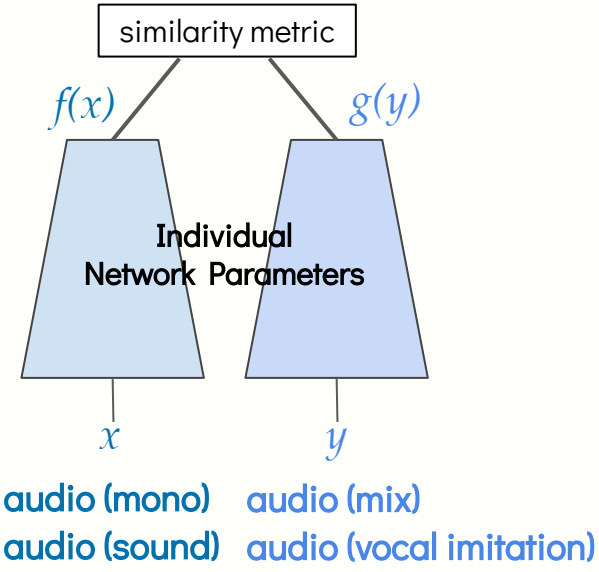
Cross-modality metric learning

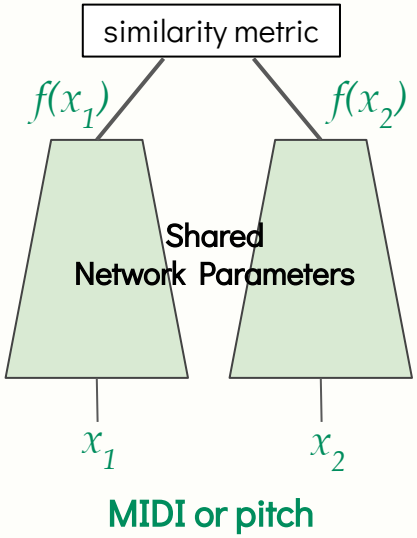
- Learning correspondence between two different modalities of inputs
  - Each modality has its own subnetwork



Cross-modality metric learning

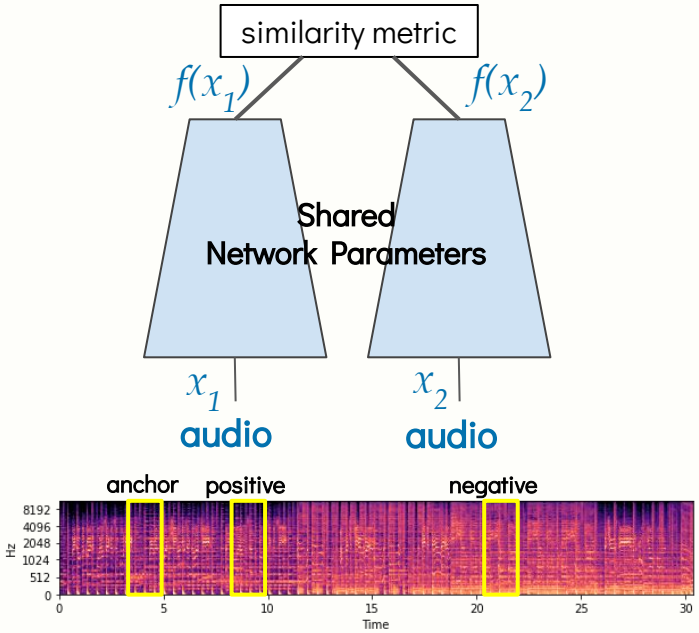
- More variations are possible
  - Different domains of audio or metric learning on symbolic data

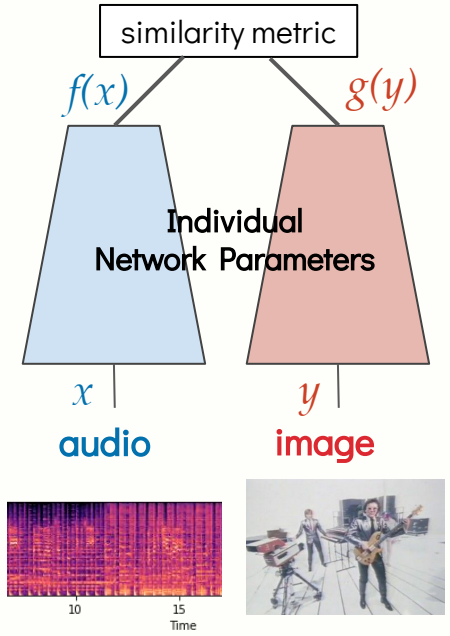




Unsupervised (or self-supervised) learning

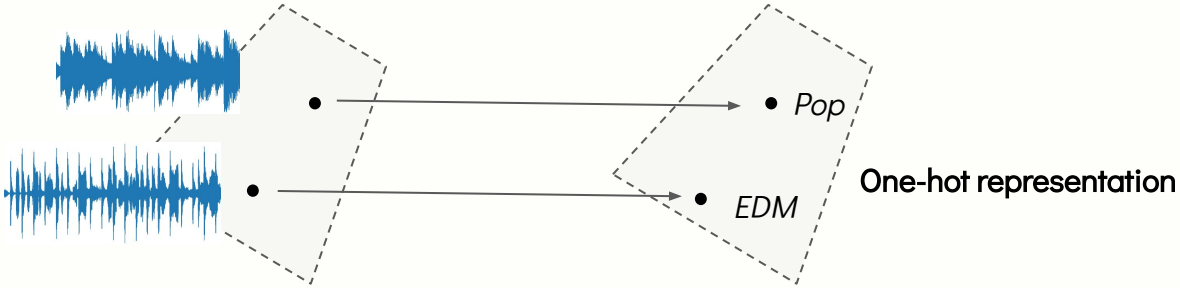
- Using **temporal proximity** in audio/music or **audio-visual correspence** in video
  - No need of manual labeling or matching





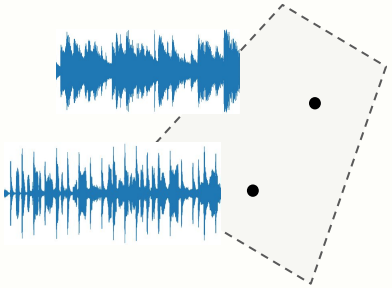
Audio and word

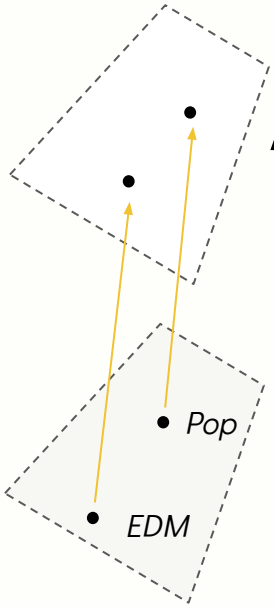
- Conventional supervised learning



Audio and word

- Learning semantic information of labels from other data sources
  - For example, regards musical instrument annotations as genre attributes: “side information”
  - Project the one-hot representation to the musical instrument embedding (attribute embedding)





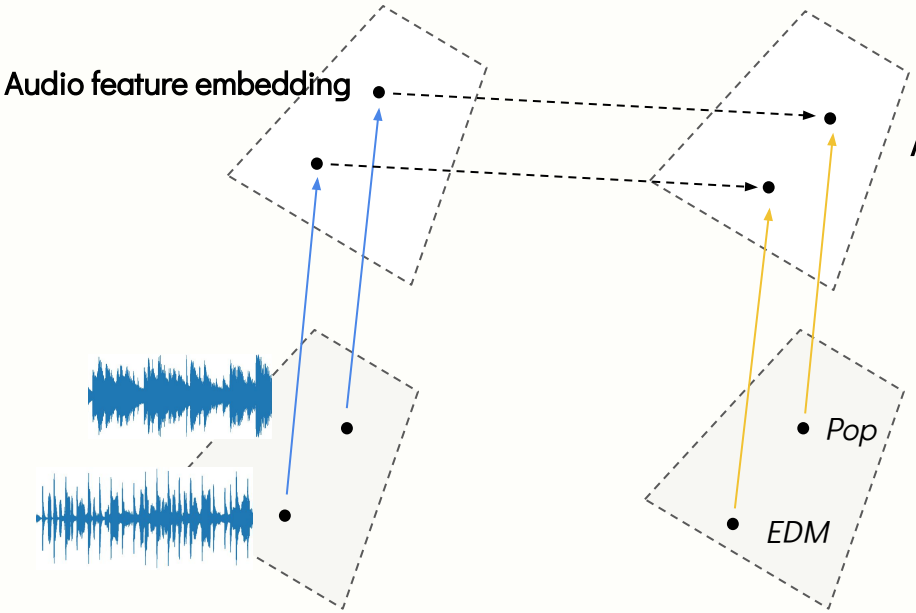
Attribute embedding

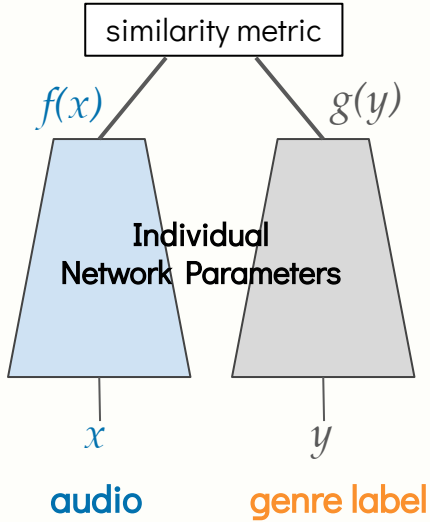
Side information: musical instrument appearances in each genre

	Drum	Bass	Guitar	Vocal	Elec drum	Synth
Pop	0.9	0.9	0.7	0.9	0.1	0.5
EDM	0.1	0.8	0.1	0.1	0.8	0.9

Audio and word

- Associates audio feature space with the attribute space using metric learning



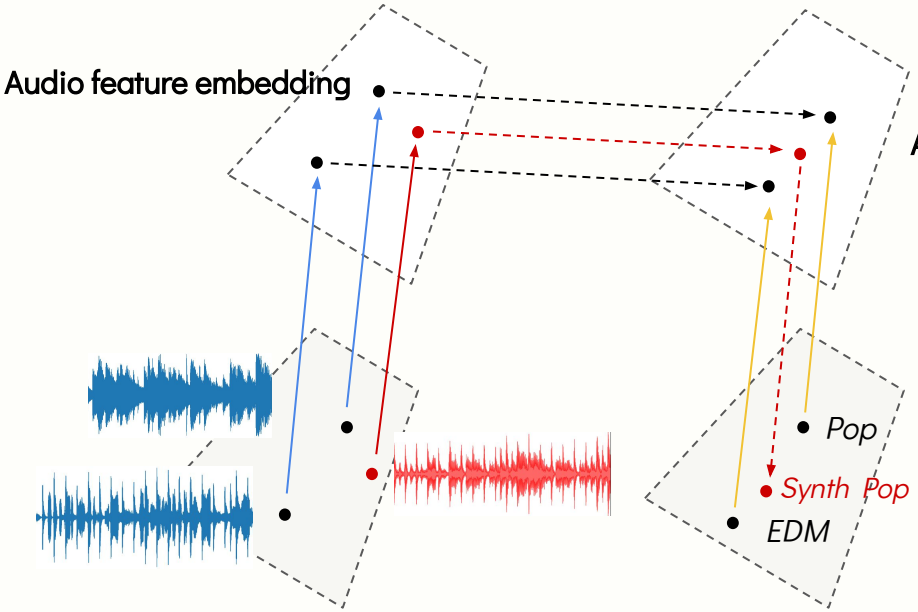


Side information: musical instrument appearances in each genre

	Drum	Bass	Guitar	Vocal	Elec drum	Synth
Pop	0.9	0.9	0.7	0.9	0.1	0.5
EDM	0.1	0.8	0.1	0.1	0.8	0.9

Audio and word: zero-shot learning

- Collect more side information than those in the training data
- This enables the model to predict unseen labels

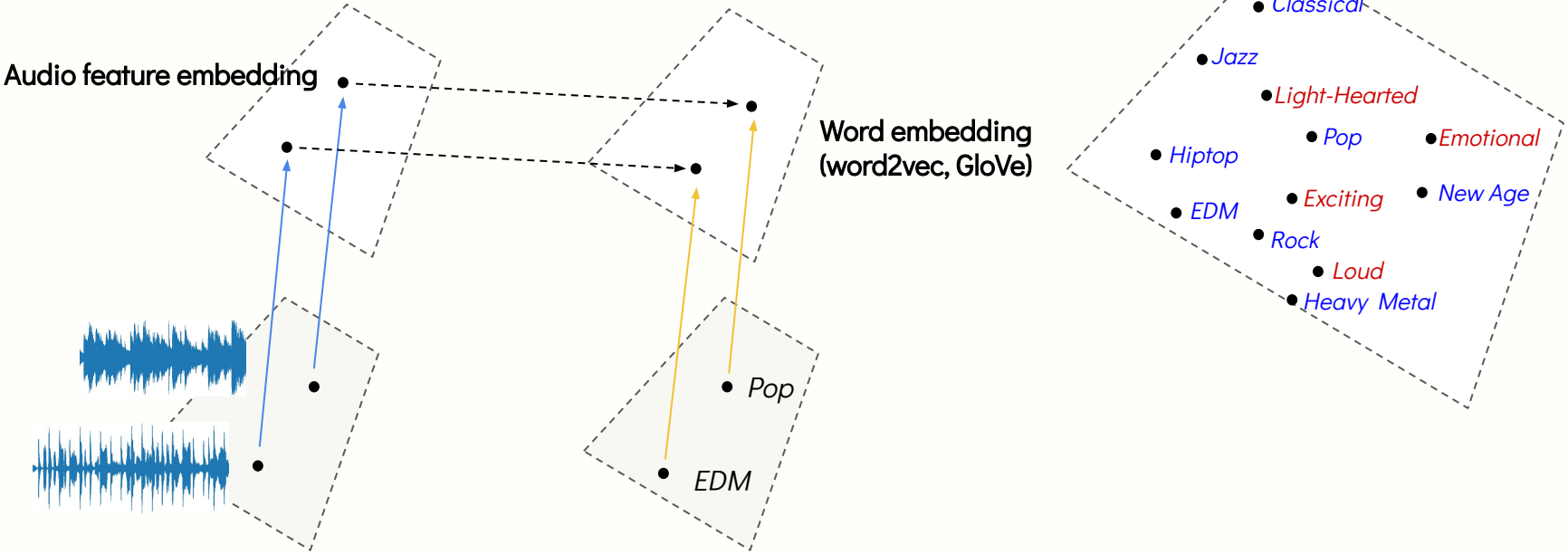


Side information: musical instrument appearances in each genre

	Drum	Bass	Guitar	Vocal	Elec drum	Synth
Pop	0.9	0.9	0.7	0.9	0.1	0.5
EDM	0.1	0.8	0.1	0.1	0.8	0.9
Synth Pop	**0.2**	**0.9**	**0.2**	**0.9**	**0.8**	**0.9**
Folk	0.5	0.3	0.9	0.9	0.0	0.1

Audio and word: zero-shot learning

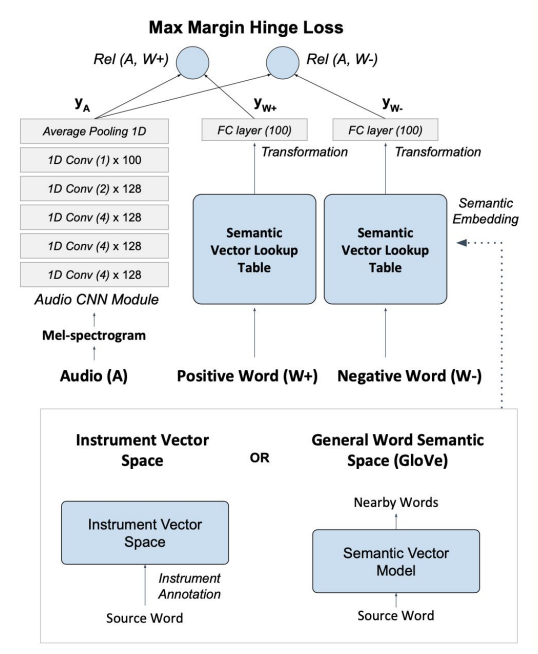
- We can also use a pre-trained word embedding
  - More rich annotations beyond music genres can be predicted!

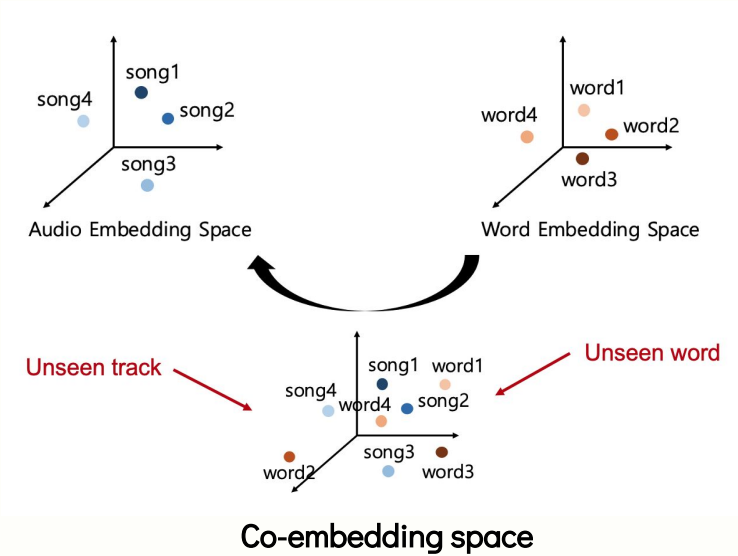


Audio and word: zero-shot learning

[Choi et al. 2019]

- Learning co-embedding between **audio** and **tags** using a triplet loss





Audio and word: zero-shot learning

[Choi et al. 2019]

- Annotation and retrieval Results

Smells like teen spirit Nirvana	Superstition Stevie Wonder	Theme to Grace / Lament George Winston
classicrock (unseen)	funk (unseen)	folk
punk	soul	instrumental
rock (unseen)	pop (unseen)	jazz
80s (unseen)	jazz	piano (unseen)
alternative	80s (unseen)	singersongwriter
punkrock	blues (unseen)	chillout
90s	classicrock	acoustic
metal	90s (unseen)	blues
vintage	disco	mellow
alternativerock	dance	chill

Top 10 auto-tagging results for examples of well- known songs including unseen tags during training

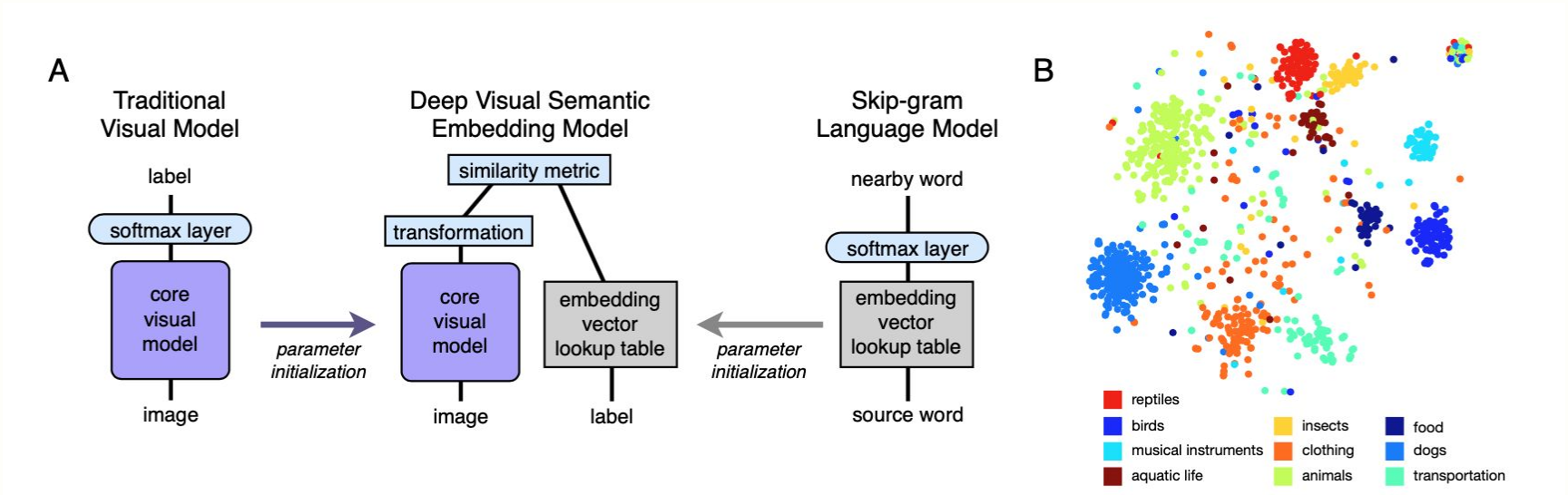
Query	Top 5 Retrieved Tracks (Title / Artist)	Original Last.fm Annotation
guitar	Iron Acton / Beak	psychedelic, experimental, krautrock, english, bass
	Drink Whiskey And Shut up / Brian Setzer	rock
	Thar She Blows / The Halibuts	party, surf, surfrock
	All Quiet On 34th Street / Eric Burdon And The New Animals	rock, rocknroll, hardrock, screamo, 00s
	Gimme Some Lovin' / Traffic	60s, british, classicrock, rock
lovely	Eddie My Love / The Chordettes	50s, doowop, oldies, pop, vocal
	Vaya Con Dios / Les Paul & Mary Ford	50s, jazz, oldies
	(I Can't Help You) I'm Falling Too / Skeeter Davis	country, oldies
	Mr. Blue / The Fleetwoods	oldies, 50s, pop, doowop, ballad
	I'm Blue Again / Patsy Cline	blue, country

Top 5 retrieved tracks for a query word from unseen tag subset ('guitar') and an arbitrary word ('lovely')

DeViSE: a deep visual-semantic embedding model

[Frome et al. 2013]

- The seminal work in the computer vision community
  - Use word embedding for labels and associate it with image embedding
  - Pretrain each subnetwork individually before cross-modal metric learning when there is semantic gap between two modalities



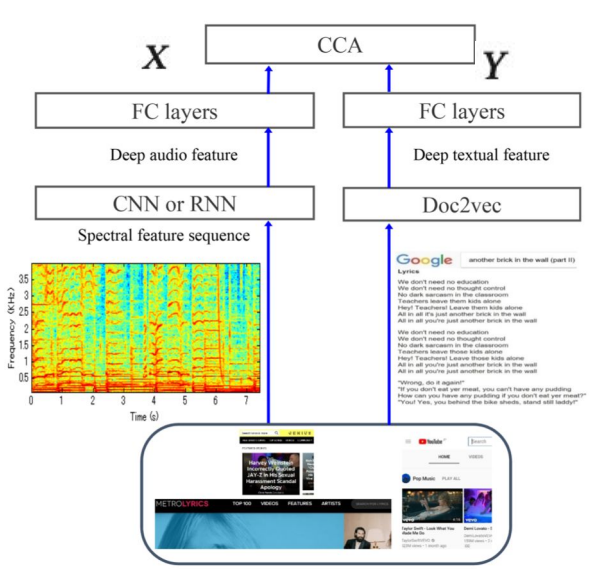
Audio and lyrics: learning the temporal correlation

[Yu et al. 2019]

- Tasks: retrieve **audio** using **lyrics** or **lyrics** using **audio**
- Summarize lyrics using “Doc2Vec”
  - Transform lyrics into a fix size of vector (300-dim) taking into account the order of words
  - 30s of audio and the corresponding lyrics are paired
- Deep Canonical Correlation Analysis (DCCA)
  - Maximize the cross-correlation between two linear projections from audio and lyrics embedding

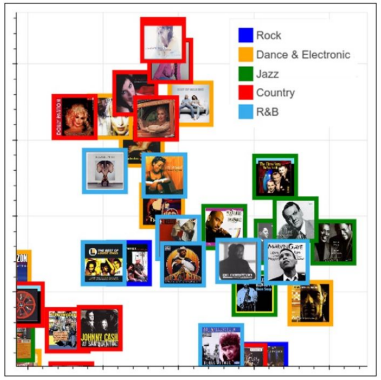
$$\underset{(H, W_X, W_Y, \varphi_x, \varphi_y)}{argmax} \; corr\left(W_X^T X, W_Y^T Y\right)$$

- RNN is more effective than CNN for audio embedding
  - Temporal order matters



Audio and album cover: genre classification

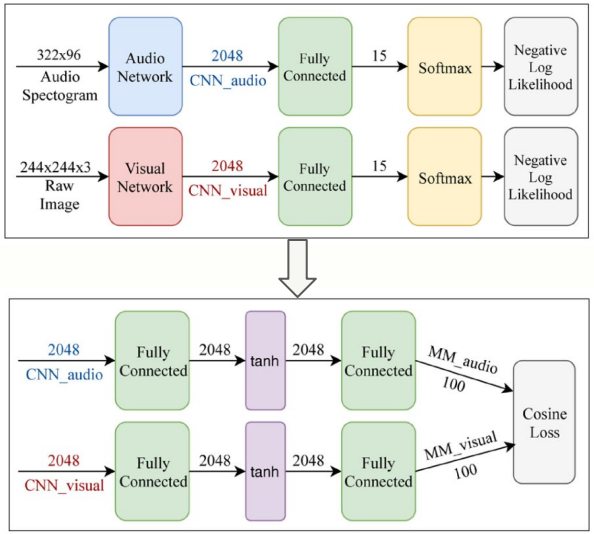
[Oramas et al. 2018]



- Learning co-embedding between **audio** and **images**
  - Audio: constant-Q transform (30s)
  - Image: album cover

- The audio and vision subnetwork are individually trained to classify genres
  - The vision subnetwork is initialized using pretrained network parameters with the ImageNet dataset

- The pretrained subnetworks are co-embedded using metric learning (triplet loss)
  - Visually or aurally-informed features are extracted for multi-label genre classification



Audio and album cover: genre classification

[Oramas et al. 2018]

- Results

Input	Model	P	R	F1
Audio	CNN_AUDIO	0.385 ± 0.006	0.341 ± 0.001	0.336 ± 0.002
	MM_AUDIO	0.406 ± 0.001	0.342 ± 0.003	0.334 ± 0.003
	CNN_AUDIO + MM_AUDIO	0.389 ± 0.005	0.350 ± 0.002	**0.346 ± 0.002**
Video	**CNN_VISUAL**	0.291 ± 0.016	0.260 ± 0.006	**0.255 ± 0.003**
	MM_VISUAL	0.264 ± 0.005	0.241 ± 0.002	0.239 ± 0.002
	CNN_VISUAL + MM_VISUAL	0.271 ± 0.001	0.248 ± 0.003	0.245 ± 0.003
A + V	CNN_AUDIO + CNN_VISUAL	0.485 ± 0.005	0.413 ± 0.005	0.425 ± 0.005
	MM_AUDIO + MM_VISUAL	0.467 ± 0.007	0.393 ± 0.003	0.400 ± 0.004
	ALL	0.477 ± 0.010	0.413 ± 0.002	**0.427 ± 0.000**

Visually-informed audio feature improves accuracy

Aurally-informed image feature does not help

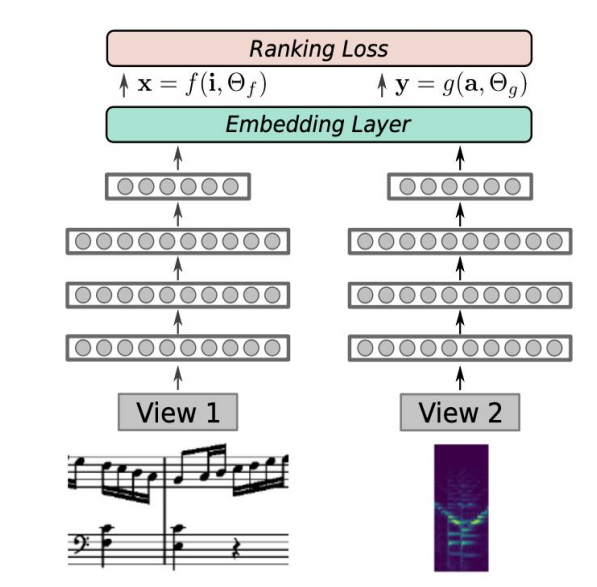
Combining all together improve accuracy further

Audio and sheet music image: short-term matching

[Dorfer et al. 2017]

- Correspond **audio** to **sheet music image**
  - Audio excerpt: 92bins x 42 frames (log-freq. spectrogram)
  - Sheet music snippet: 180 x 200 pixels
- Data augmentation
  - Audio: synthesized with different samples and tempo change
  - Sheet music: scaling, translation
- Use the triplet hinge loss
  - Cosine similarity
  - Anchor: sheet music embedding
  - Positive: matching audio
  - Negative: non-matching audio

$$\mathcal{L}_{rank} = \sum_{\mathbf{x}} \sum_{k} \max\{0, \alpha - s(\mathbf{x}, \mathbf{y}) + s(\mathbf{x}, \mathbf{y}_k)\}$$

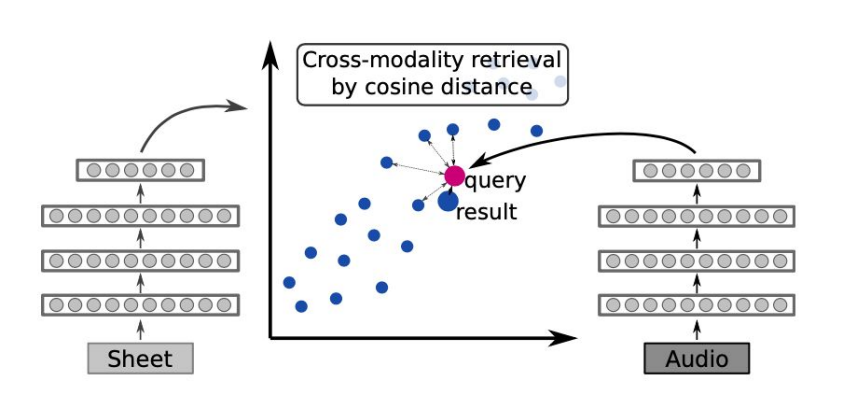


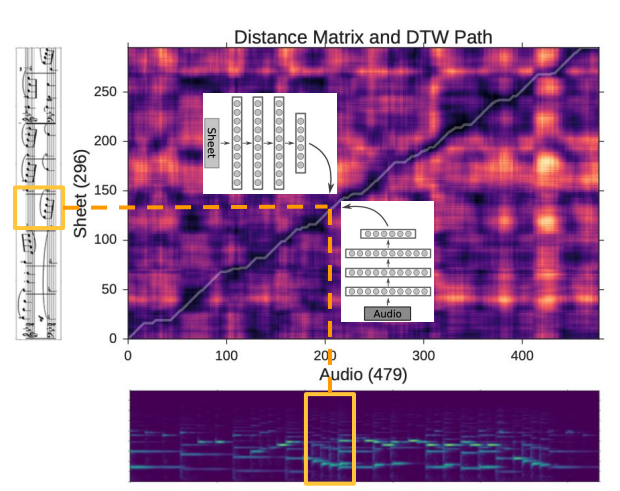
sheet music snippet audio excerpt

Audio and sheet music image: short-term matching

[Dorfer et al. 2017]

- Task 1: sheet music identification from audio queries: voting of retrieved results from all audio excerpts within a whole audio recording
- Task 2: offline alignment of a given audio with the sheet music image using the distance matrix where each point is computed from the two embeddings



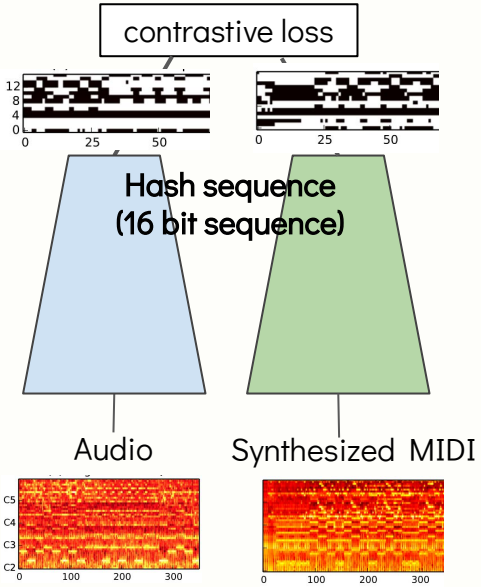


Audio and MIDI: cross-modality hashing

[Raffel and Ellis 2015]

- Large-scale content-based **MIDI**-to-**audio** retrieval
  - Find matching between 140,910 MIDI files and 994,950 audio files
  - The traditional method (DTW over CQT/chroma) is too slow!
  - Computing the distance matrix is the bottleneck

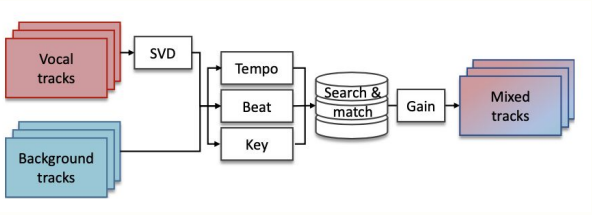
- Speed-up by learning “binary vectors” from audio and synthesized MIDI and computing hamming distance
  - Use a CNN-based Siamese network (CNN) and the contrastive loss
  - The exclusive-or operation (hamming distance) between the hash sequences is 400 times faster than the inner product between the CQT features
  - The DTW score with the hamming distance is 9 times faster



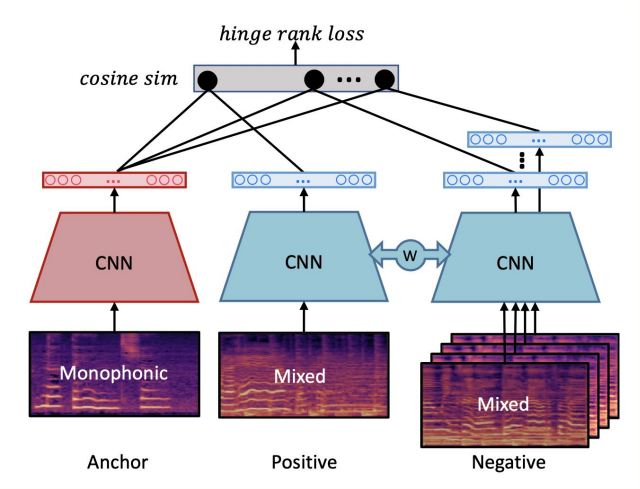
Audio-to-audio: mono and mix matching

[Lee and Nam 2019]

- Singer identification regardless of the presence of background music
- Learn co-embedding space between **monophonic audio** and its **mixed version**
- Data generation
  - Vocal track: karaoke singing (DAMP)
  - Mixed track: auto-mashup between the vocal and background tracks (musdb18) using tempo, beat, key matching



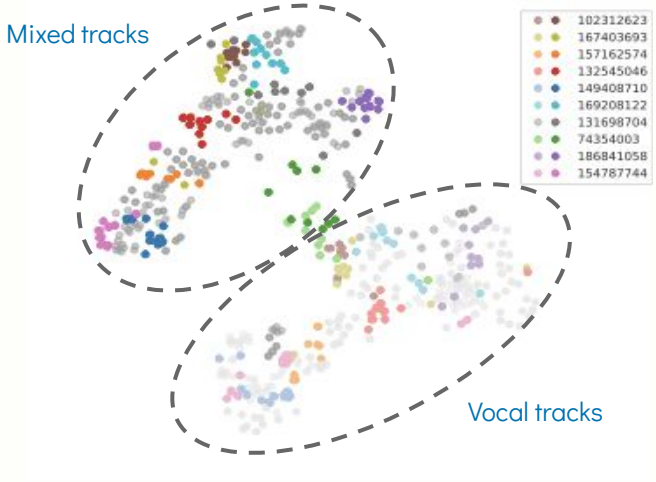
- Use the triplet hinge loss with cosine similarity



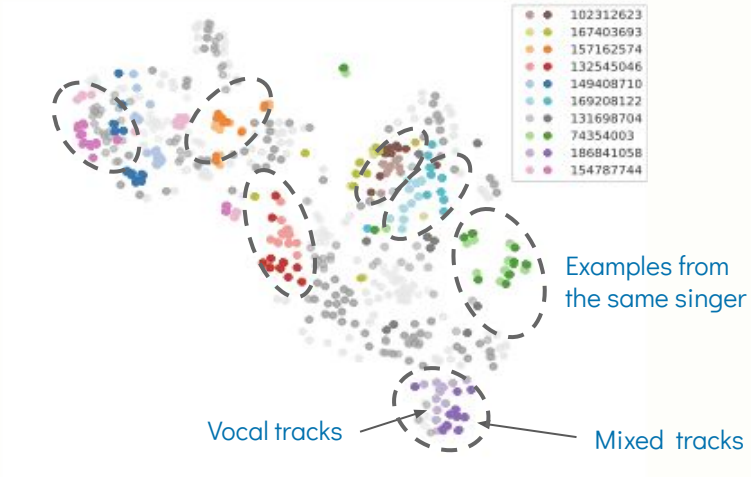
Audio-to-audio: mono and mix

[Lee and Nam 2019]

- Vocal tracks and their mixed tracks are projected closely: “implicit vocal source separation”



Before cross-modality metric learning
(the model is trained with mixed tracks only)



After cross-modality metric learning
(the model is jointly trained with mono and mixed tracks)

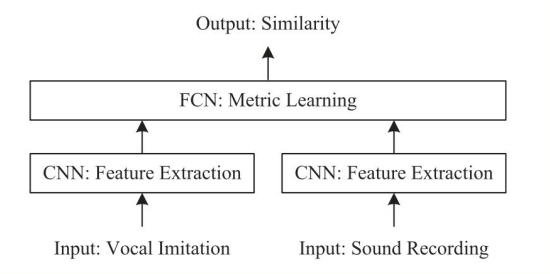
Audio-to-audio: sound sample and vocal imitation

[Zhang et al. 2019]

- Retrieve **sound samples** using **vocal imitation as query**
  - E.g.: electronic/mechanical failure sounds imitated by callers in the NPR Car Talk show

- The VocalSketch dataset
  - Crowdsourced 10 vocal imitations of a target sound
  - Everyday sounds, acoustic instrument, synthesizers

- Siamese-style network: trained to classify into paired or not paired
  - Combine the two embeddings and do the binary classification (use the categorical cross-entropy loss)



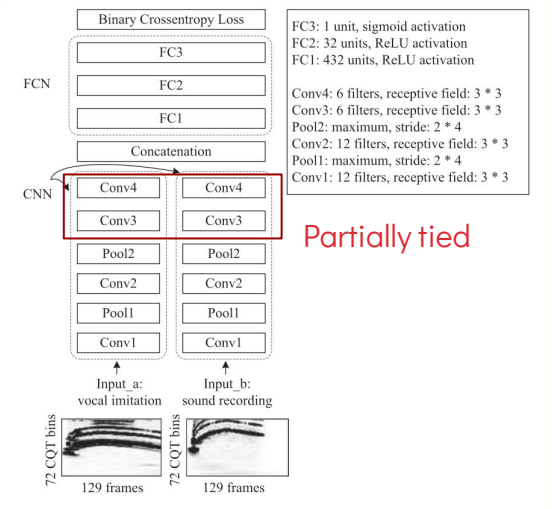
Audio-to-audio: sound sample and vocal imitation

[Zhang et al. 2019]

- Symmetric model (IMINET)
  - Two subnetworks have the same configuration
  - Network parameters (weights) are shared in different settings: tied, partially tied or untied,
  - Assuming that the two modalities share certain high-level characteristics to recognize sounds

- Results
  - Tied setting works best but the combined model is even better

Configuration	Acoustic Instr.	Comm. Synthesizers	Everyday	Single Synthesizer
Untied	0.377 ± 0.019	0.318 ± 0.020	0.154 ± 0.014	0.325 ± 0.020
Partial	0.384 ± 0.027	0.304 ± 0.015	0.154 ± 0.015	0.340 ± 0.031
Tied	0.401 ± 0.028	0.327 ± 0.019	0.158 ± 0.012	0.380 ± 0.018
Untied + Partial + Tied	**0.438 ± 0.015**	**0.343 ± 0.020**	**0.175 ± 0.012**	**0.382 ± 0.013**



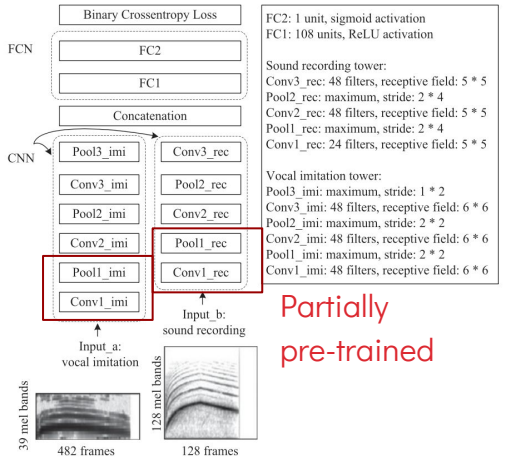
Symmetric model

Audio-to-audio: sound sample and vocal imitation

[Zhang et al. 2019]

- Asymmetric model (TL-IMINET)
  - Each subnet is trained independently
    - The vocal imitation subnetwork: to classify 7 different languages
    - The sound subnetwork: to classify 10 environmental sounds
  - The pre-trained models are transferred to the Siamese model with a different amount
    - Conv1, Conv1/2, or Conv 1/2/3
- Results
  - The fully pre-trained model (Conv 1/2/3) works best

Configuration	Acoustic Instr.	Comm. Synthesizers	Everyday	Single Synthesizer
No Pre-train	0.397 ± 0.027	0.309 ± 0.021	0.225 ± 0.023	0.377 ± 0.025
Pre-train Conv1	0.412 ± 0.033	0.328 ± 0.027	0.227 ± 0.020	0.399 ± 0.036
Pre-train Conv1/2	0.432 ± 0.024	0.325 ± 0.023	0.225 ± 0.016	**0.404 ± 0.036**
Pre-train Conv1/2/3	**0.462 ± 0.017**	**0.349 ± 0.015**	**0.246 ± 0.016**	0.390 ± 0.027

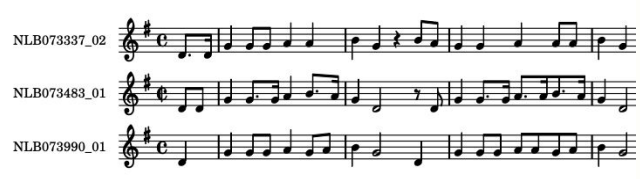


Asymmetric model

MIDI-to-MIDI: symbolic melody similarity

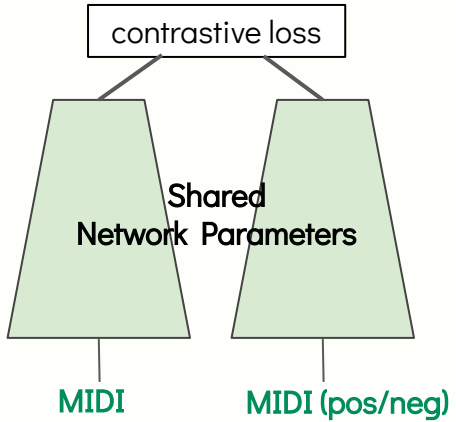
[Karsdorp et al. 2019]

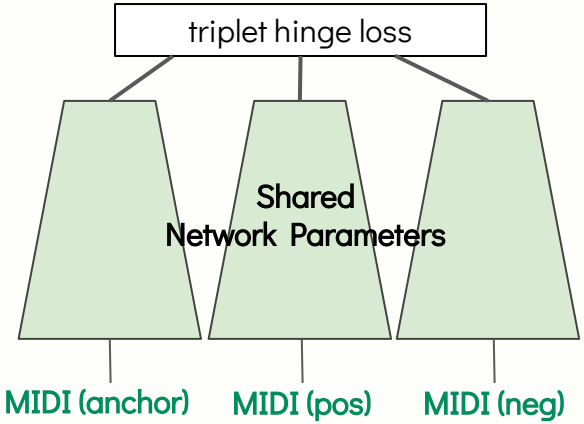
- Symbolic representation learning for melody retrieval
  - MTC-FS-INST 2.0 (Meetens Tune Collection) dataset
  - 9, 336 melodies in 2,094 tune families



Three variant melodies from a tune family

- MIDI is represented as a sequence of note events
  - Each note event contains pitch, duration, beat and so on
  - Use RNN to takes the entire note event sequence as input
- Two metric learning models
  - Siamese model and triplet loss model

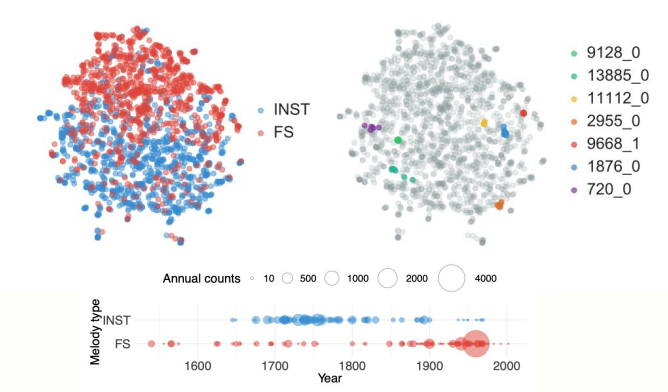




MIDI-to-MIDI: symbolic melody similarity

[Karsdorp et al. 2019]

- Visualization of the melody embedding space
  - Instrument melody and vocal melody are well-separated (They are from different years)
  - The same tune families are closely clustered



- Tune identification result
  - The Siamese model (duplet) is more effective than the triplet model
  - The RNN-based embedding is much faster than the alignment-based approach

	MAP			P@1	Sil.
	all	seen	unseen		
RNN_D	0.72	0.70	0.74	0.78	0.33
RNN_T	0.68	0.64	0.71	0.75	0.28
Alignment	0.67	–	–	0.78	0.22

Pitch-to-pitch: cover song detection

[Doras and Peeters 2019]

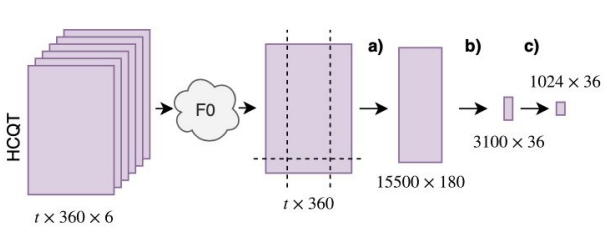


Figure 2: Input data pre-processing: dominant melody is extracted from HCQT, then a) F0 output is trimmed, b) downsampled by a factor 5 and c) resized time-wise to 1024 bins (time bins on first dimension, frequency bins on second dimension).

- Use a dominant pitch estimation model as the front-end processing
  - Summarize the pitch sequence with trimming, downsampling and resizing,
  - Represent a track with a fixed size of 1024 (time) x 36 (pitch) matrix
  - The SecondHandSongs dataset for cover song detection

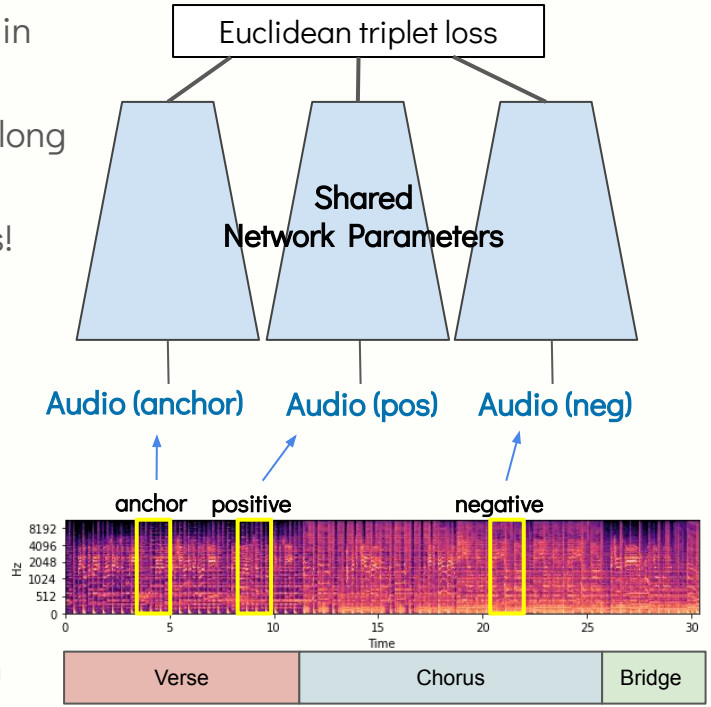
- Triplet loss model (CNN)
  - Take the track-level pitch matrix as input : no need of the alignment between two tracks
  - Online semi-hard negative pair mining
    - For each of the positive pair, if any negative pair distance is closer than the positive distance, only the negative with highest distance is kept.
    - If there is no such negative pairs, only the negative pair with lowest distance is kept.

Unsupervised learning: music segmentation

[McCallum 2019]

- Learn audio features for music segmentation
  - Exploit the time proximity of audio features that is implicit in any audio timeline.
  - Two neighboring audio segments in music are likely to belong to the same functional group such as verse or chorus
  - Possible to learn the features without segmentation labels!

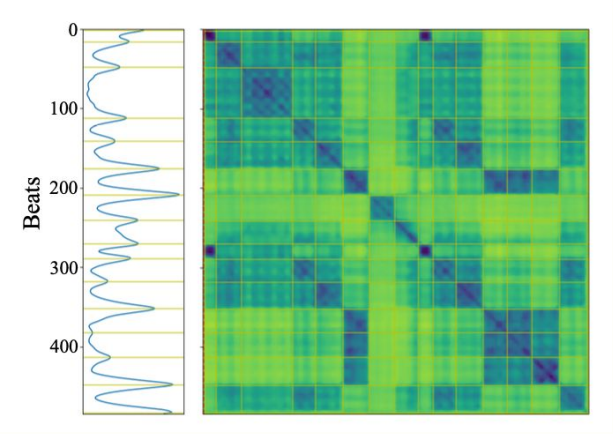
- Triplet model (CNN)
  - Audio input: beat-sync CQT
    - Input size are “snapped” on beats
    - Save the model from learning tempo-invariance
  - Biased sampling: use a signal-level distance (2D FFT log-magnitude over beat-synced CQT) between the anchor and positive to reduce false selection



Unsupervised learning: music segmentation

[McCallum 2019]

- Results
  - Beat-synchronized input improves performance
  - Biased sampling reduces false selection of positive examples



Detected boundaries (yellow) and Self-Similarity Matrix (SSM) from the learned embedding

Table 1. Performance metrics for the BeatlesTUT dataset

Algorithm	F-Measure	Precision	Recall
[4]	0.503 ± 0.18	0.579 ± 0.21	0.461 ± 0.17
[6]	0.651 ± 0.17	0.622 ± 0.19	**0.708** ± 0.19
Unsynchronized	0.597 ± 0.17	0.589 ± 0.19	0.625 ± 0.17
Beat-Synchronized	0.648 ± 0.17	0.647 ± 0.20	0.677 ± 0.18
Biased Sampling	**0.662** ± 0.17	**0.663** ± 0.20	0.691 ± 0.19

Table 2. Performance metrics for the SALAMI-A dataset

Algorithm	F-Measure	Precision	Recall
[4]	0.446 ± 0.17	0.457 ± 0.21	0.483 ± 0.19
[6]	0.493 ± 0.17	0.454 ± 0.20	0.595 ± 0.19
Unsynchronized	0.497 ± 0.16	0.429 ± 0.18	0.653 ± 0.15
Beat-Synchronized	**0.535** ± 0.15	**0.491** ± 0.20	**0.660** ± 0.16
Biased Sampling	0.533 ± 0.16	**0.491** ± 0.21	0.656 ± 0.16

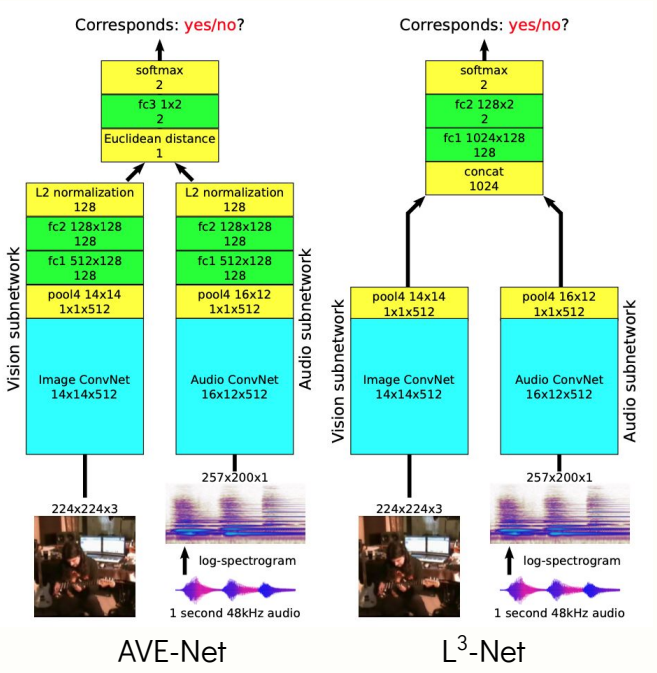
Music segmentation accuracy

Unsupervised learning: audio-visual correspondence

[Arandjelović and Zisserman, 2018]

- Use unlabeled video data
  - Given an input pair of a video frame and 1s audio, the model is trained to decide if it is from the same position
  - It is called **"self-supervision"** because the labels are constructed directly from the data itself
  - Virtually unlimited paired data!

- The audio and image embeddings are combined and classified into paired or not
  - L^3-Net (Look, Listen and Learn): concatenation
  - AVE-Net: Euclidean distance
  - AudioSet: a large collection of YouTube video clips (10s)
  - The entire networks were trained from scratch

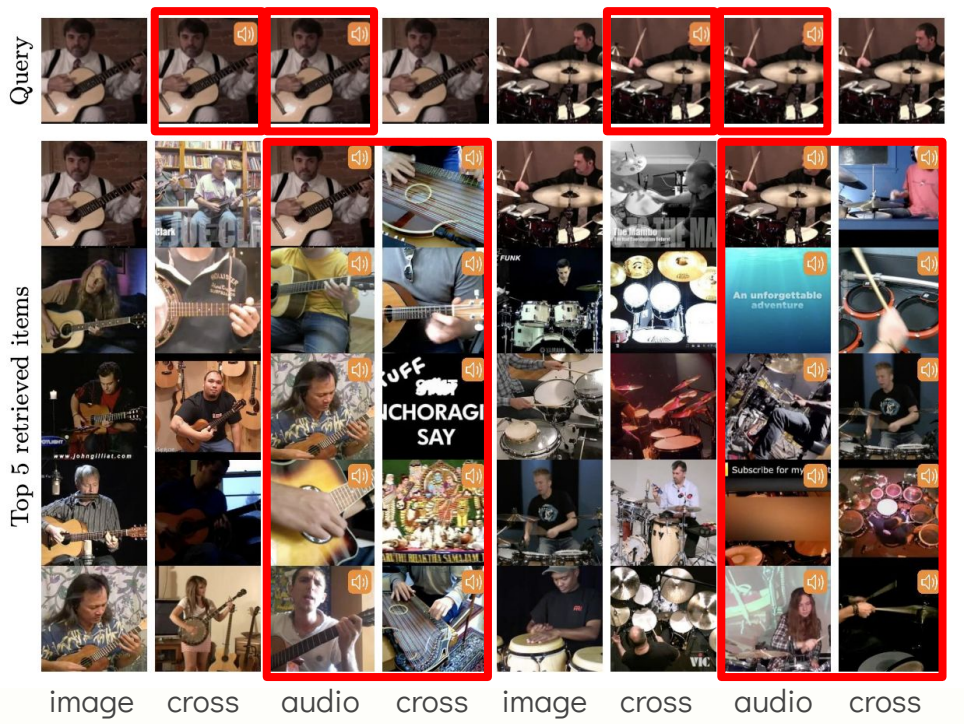


Unsupervised learning: audio-visual correspondence

[Arandjelović and Zisserman, 2018]

- Cross-modal (audio-image) and intra-modal (audio-audio, image-image) retrieval

audio only



Summary of part 3

- “Music data” include various modalities in the form of audio, image, MIDI or text
- Deep learning allows modality-agnostic representation learning
- Deep metric learning provides a flexible way of matching the different modalities of music data in a supervised or unsupervised way, from which we can learn the joint embedding space
- It is often effective to pretrain each subnetwork individually, if available, using other datasets (i.e., transfer learning) before cross-modality metric learning
- There are a lot of use cases!

Closing remarks: leverage more music data correspondences!

- Playlist (tracks, title, image)
- User data (listening history) and tracks
- Movie/game/drama: scenes and background music (mood)
- Music performance videos: performers' motion and instrumental sound

Closing remarks: trends in deep learning

- Unsupervised (or self-supervised) learning
  - Free metadata (artist/album/track) [Lee et al. 2019] or free labels (audio-visual correspondence)
  - Exploit common nature or structure of music and audio: temporal proximity, category preservation of sounds in its mixture with other sources [Jansen et al. 2018]
  - Audio effects (pitch shifting [Gfeller et al. 2020]) or musical transform (key transpose, tempo changes?)

- Semi-supervised learning
  - Use unlabeled data with data augmentation in metric learning settings

- Disentanglement of embedding space
  - Multi-level conditions (e.g., genre + tempo + instrument) is possible
  - Hierarchical disentanglement and masking

Closing remarks: open problems

- Large-scale retrieval between sequential data (e.g., cover song, midi-to-audio)
- Metric learning with score-level similarity: chord progress, rhythm pattern
- Polysemy problem
- Applications to music generation: e.g., similarity distribution for generated music

Thank you!

The Github website:
https://github.com/bmcfee/ismir2020-metric-learning

References

- [Choi et al. 2019] Jeong Choi, Jongpil Lee, Jiyoung Park, and Juhan Nam, **Zero-shot learning for audio-based music classification and tagging**, ISMIR 2019
- [Frome et al. 2013] Greg S. Corrado, Jonathon Shlens, Samy Bengio, Jeffrey Dean, Marc'Aurelio Ranzato, and Tomas Mikolov, **DeViSE: A Deep Visual-Semantic Embedding Model, Andrea Frome**, NIPS 2013
- [Yu et al. 2019] Yi Yu, Suhua Tang, Francisco Raposo, and Lei Chen, **Deep Cross-Modal Correlation Learning for Audio and Lyrics in Music Retrieval**, ACM Trans. Multimedia Comput. Commun. Appl. Vol. 15, No. 1, 2019
- [Oramas et al. 2018] Sergio Oramas, Francesco Barbieri, Oriol Nieto, and Xavier Serra, **Multimodal Deep Learning for Music Genre Classification**, TISMIR 2018
- [Dorfer et al. 2017] Matthias Dorfer, Andreas Arzt, Gerhard Widmer, **Learning Audio-Sheet Music Correspondences for Score Identification and Offline Alignment**, ISMIR 2017
- [Raffel and Ellis 2015] Colin Raffel, Daniel P. W. Ellis, **Large-Scale Content-Based Matching of MIDI and Audio Files**, ISMIR 2015

References

- [Lee and Nam 2019] Kyungyun Lee and Juhan Nam, **Learning a Joint Embedding Space of Monophonic and Mixed Music Signals for Singing Voice**, ISMIR 2019
- [Zhang et al. 2019] Yichi Zhang, Bryan Pardo, Zhiyao Duan, **Siamese Style Convolutional Neural Networks for Sound Search by Vocal Imitation**, IEEE TASLP 2019
- [Karsdorp et al. 2019] Folgert Karsdorp, Peter Kranenburg, and Enrique Manjavacas, **Learning Similarity Metrics for Melody Retrieval**, ISMIR 2019
- [Doras and Peeters 2019] Guillaume Doras and Geoffroy Peeters, **Cover Detection using Dominant Melody Embeddings**, ISMIR 2019
- [McCallum 2019] Matthew C. McCallum, **Unsupervised Learning of Deep Features for Music Segmentation**, ICASSP, 2019
- [Arandjelović and Zisserman, 2018] Relja Arandjelović and Andrew Zisserman, **Objects that Sound**, ECCV 2018

References

- [Gfeller et al. 2020] Beat Gfeller, Christian Frank, Dominik Roblek, Matt Sharifi, Marco Tagliasacchi, Mihajlo Velimirović, **SPICE: Self-supervised Pitch Estimation**, IEEE TASLP 2020
- [Jansen et al. 2018] Aren Jansen, Manoj Plakal, Ratheet Pandya, Daniel P. W. Ellis, Shawn Hershey, Jiayang Liu, R. Channing Moore, Rif A. Saurous, **Unsupervised Learning of Semantic Audio Representations**, ICASSP 2018
- [Lee et al. 2019] Jongpil Lee, Jiyoung Park, and Juhan Nam, **Representation Learning of Music Using Artist, Album, and Track information**, ML4MD, ICML 2019