An end-to-end approach for the verification problem: learning the right distance

João Monteiro[1,2], Isabela Albuquerque[1], Jahangir Alam[1,2], R Devon Hjelm[3,4], Tiago H. Falk[1]

1-Institut National de la Recherche Scientifique (INRS-EMT)
2-Centre de Recherche Informatique de Montréal (CRIM)
3-Microsoft Research
4-Quebec Artificial Intelligence Institute (MILA)

Outline

- **Background**
  - The verification problem
  - Distance metric learning / Metric learning

- Learning pseudo metric spaces
  - TL;DR
  - Method
  - Main results
  - Training details

- Evaluation
  - Verifying standard distance properties in trained models
  - Proof-of-concept experiments on images
  - Open-set speaker verification

The verification problem

- Given a *trial* $T=\{x_1, x_2\}$, decide whether the underlying classes are the same (*target trial*) or not (*non-target trial*)
  - Trial: a pair of examples (or a pair of sets of examples)

The verification problem

- Given a *trial* $T=\{x_1, x_2\}$, decide whether the underlying classes are the same (*target trial*) or not (*non-target trial*)
  - Trial: a pair of examples (or a pair of sets of examples)

- Two settings:
  - Closed-set:
    - Same classes at train and test time
  - Open-set:
    - New classes at test time

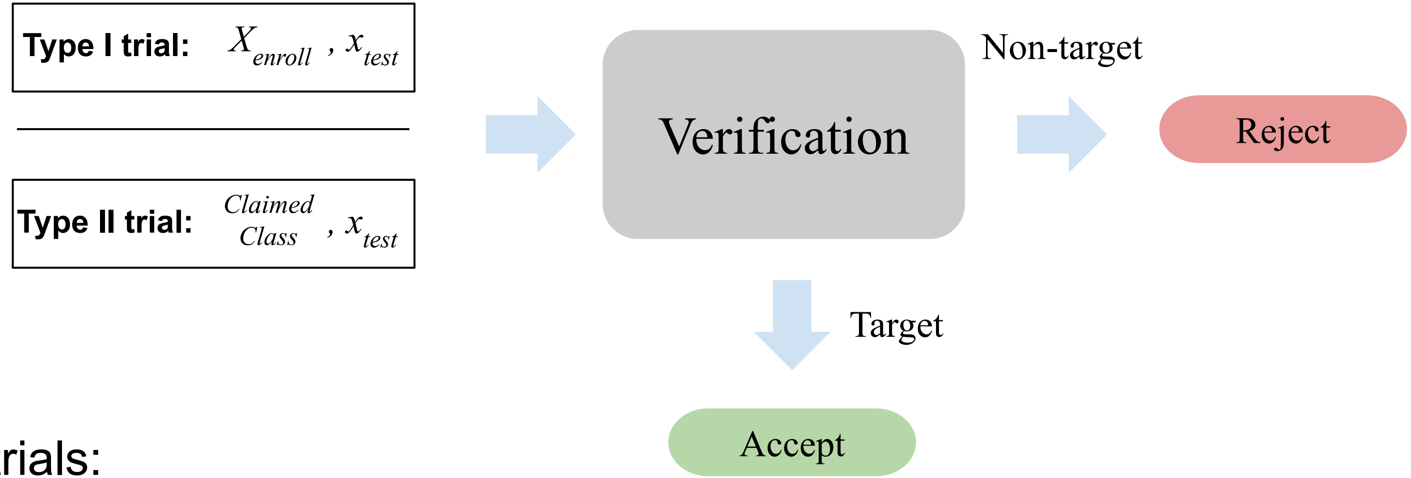
The verification problem

- Given a *trial* $T=\{x_1, x_2\}$, decide whether the underlying classes are the same (*target trial*) or not (*non-target trial*)
  - Trial: a pair of examples (or a pair of sets of examples)

- Two settings:
  - Closed-set:
    - Same classes at train and test time
  - Open-set:
    - New classes at test time

- Popular instances:
  - Biometrics
  - Forensics

The verification problem



- Type I trials:
  - Enrollment set + test example

- Type II trials:
  - Claimed class + test example
  - Closed-set only

The Neyman-Pearson approach to the verification problem

$$LR = \frac{p(T|H_0)}{p(T|H_1)}$$

- H_0: Target trials (same classes)
- H_1: Non-target trials (different classes)
- Decision rule: Compare the likelihood ratio (LR) with a threshold

The Neyman-Pearson approach to the verification problem

$$LR = \frac{p(T|H_0)}{p(T|H_1)} \quad \Rightarrow \quad LR = \frac{p_{X_{Enroll}}(x_{test})}{p_{UBM}(x_{test})}$$

- H_0: Target trials (same classes)
- H_1: Non-target trials (different classes)
- Decision rule: Compare the likelihood ratio (LR) with a threshold
- Generative approaches approximate both terms in LR
  - Very often employing complex pipelines
  - Some attempts towards end-to-end settings in recent literature

Distance metric learning / Metric learning

- Represent data in a metric space where distances indicate semantic relationships

Distance metric learning / Metric learning

- Represent data in a metric space where distances indicate semantic relationships
  - Distance metric learning: learn how to assess similarity/distance
    - E.g., Mahalanobis distance learning (Xing et al. 2003):

      Learn A s.t. $\sqrt{(x-y)^t A(x-y)}$ is small for semantically close x and y, where A is positive semi-definite.

Distance metric learning / Metric learning

- Represent data in a metric space where distances indicate semantic relationships
  - Distance metric learning: learn how to assess similarity/distance
    - E.g., Mahalanobis distance learning (Xing et al. 2003):

      Learn A s.t. $\sqrt{(x-y)^t A(x-y)}$ is small for semantically close x and y, where A is positive semi-definite.
  - Metric learning: learn an encoding process instead
    - E.g., Siamese nets (Bromley et al. 1994, Chopra et al. 2005, Hadsell et al. 2006):

      Learn a mapping $\mathcal{E}$ s.t. $||\mathcal{E}(x) - \mathcal{E}(y)||_2$ is small for semantically close x and y.

Outline

- Background
  - The verification problem
  - Distance metric learning / Metric learning

- **Learning pseudo metric spaces**
  - TL;DR
  - Method
  - Main results
  - Training details

- Evaluation
  - Verifying standard distance properties in trained models
  - Proof-of-concept experiments on images
  - Open-set speaker verification

TL;DR

- Simultaneously learn the encoding process and a (pseudo) distance
  - Get a (pseudo) metric space tailored to the task at hand
  - Approximate the density ratio commonly used for hypothesis tests under generative verification

- From a practical perspective:
  - Simplify training compared to standard metric learning
  - End-to-end scoring as opposed to complex verification pipelines

Method

- Learn encoder and “distance” such that:

$$\mathcal{E}, \mathcal{D} = \arg\min \; -\mathbb{E}_{x^+ \sim p^+} \log(\mathcal{D} \circ \mathcal{E}(x^+)) - \mathbb{E}_{x^- \sim p^-} \log(1 - \mathcal{D} \circ \mathcal{E}(x^-))$$

x^+ : Positive pair of examples (same class)

x^- : Negative pair of examples

$$\mathcal{D} \circ \mathcal{E}(x^+) = \mathcal{D}(\mathcal{E}(x^+))$$

Method

- Learn encoder and “distance” such that:

$$\mathcal{E}, \mathcal{D} = \arg\min \; -\mathbb{E}_{x^+ \sim p^+} \log(\mathcal{D} \circ \mathcal{E}(x^+)) - \mathbb{E}_{x^- \sim p^-} \log(1 - \mathcal{D} \circ \mathcal{E}(x^-))$$

$\mathcal{D}$ discriminates encoded positive and negative pairs of examples

Main results

$$\mathcal{E}, \mathcal{D} = \arg\min \; -\mathbb{E}_{x^+ \sim p^+} \log(\mathcal{D} \circ \mathcal{E}(x^+)) - \mathbb{E}_{x^- \sim p^-} \log(1 - \mathcal{D} \circ \mathcal{E}(x^-))$$

- It is well known that the optimal discriminator will yield the density ratio:

$$\mathcal{D}^*(z') = \frac{p_z^+(z')}{p_z^+(z') + p_z^-(z')}$$

Main results

$$\mathcal{E}, \mathcal{D} = \text{arg min } -\mathbb{E}_{x^+ \sim p^+} \log(\mathcal{D} \circ \mathcal{E}(x^+)) - \mathbb{E}_{x^- \sim p^-} \log(1 - \mathcal{D} \circ \mathcal{E}(x^-))$$

- It is well known that the optimal discriminator will yield the density ratio:

$$\mathcal{D}^*(z') = \frac{p_z^+(z')}{p_z^+(z') + p_z^-(z')}$$

- And we have the following for trials such that $T = \{x_{enroll}, x_{test}\}$:

$$\frac{p_z^+(z')}{p_z^-(z')} = \frac{p_z^+(\mathcal{E}(x_{enroll}), \mathcal{E}(x_{test}))}{p_z^-(\mathcal{E}(x_{enroll}), \mathcal{E}(x_{test}))} := \frac{p(T|H_0)}{p(T|H_1)}$$

Main results

$$\mathcal{E}, \mathcal{D} = \arg\min \; -\mathbb{E}_{x^+ \sim p^+} \log(\mathcal{D} \circ \mathcal{E}(x^+)) - \mathbb{E}_{x^- \sim p^-} \log(1 - \mathcal{D} \circ \mathcal{E}(x^-))$$

- For the encoder, we plug the optimal discriminator into the above and find that:

$$\mathcal{E}^* \Rightarrow \; supp(p_z^+) \cap supp(p_z^-) = \varnothing$$

Main results

$$\mathcal{E}, \mathcal{D} = \arg\min \; -\mathbb{E}_{x^+ \sim p^+} \log(\mathcal{D} \circ \mathcal{E}(x^+)) - \mathbb{E}_{x^- \sim p^-} \log(1 - \mathcal{D} \circ \mathcal{E}(x^-))$$

- For the encoder, we plug the optimal discriminator into the above and find that:

$$\mathcal{E}^* \Rightarrow \; supp(p_z^+) \cap supp(p_z^-) = \varnothing$$

- The density ratio given by the optimal discriminator and encoder is calibrated in the sense that selecting a threshold is trivial:
  - The ratio will always explode or collapse
  - Any positive threshold yields correct decisions

Training details

Algorithm 1 Training procedure.

$\mathcal{E}, \mathcal{D} = InitializeModels()$
repeat
  $x, y = SampleMinibatch()$
  $z = \mathcal{E}(x)$
  $z^+ = GetAllPositivePairs(z, y)$
  $z^- = GetAllNegativePairs(z, y)$
  $y' = ProjectOntoSimplex(z)$
  $\mathcal{L}' = \mathcal{L}(z^+, z^-) + \mathcal{L}_{CE}(y', y)$
  $\mathcal{E}, \mathcal{D} = UpdateRule(\mathcal{E}, \mathcal{D}, \mathcal{L}')$
until Maximum number of iterations reached
return $\mathcal{E}, \mathcal{D}$

- Training can be carried out with alternate or simultaneous updates
  - We found both to perform similarly

Training details

Algorithm 1 Training procedure.

$\mathcal{E}, \mathcal{D} = InitializeModels()$
repeat
  $x, y = SampleMinibatch()$
  $z = \mathcal{E}(x)$
  $z^+ = GetAllPositivePairs(z, y)$
  $z^- = GetAllNegativePairs(z, y)$
  $y' = ProjectOntoSimplex(z)$
  $\mathcal{L}' = \mathcal{L}(z^+, z^-) + \mathcal{L}_{CE}(y', y)$
  $\mathcal{E}, \mathcal{D} = UpdateRule(\mathcal{E}, \mathcal{D}, \mathcal{L}')$
until Maximum number of iterations reached
return $\mathcal{E}, \mathcal{D}$

- Training can be carried out with alternate or simultaneous updates
  - We found both to perform similarly
- We make further use of labels to compute a standard classification loss
  - Found empirically to accelerate training

Training details

Algorithm 1 Training procedure.

$\mathcal{E}, \mathcal{D} = InitializeModels()$
repeat
  $x, y = SampleMinibatch()$
  $z = \mathcal{E}(x)$
  $z^{+} = GetAllPositivePairs(z, y)$
  $z^{-} = GetAllNegativePairs(z, y)$
  $y' = ProjectOntoSimplex(z)$
  $\mathcal{L}' = \mathcal{L}(z^{+}, z^{-}) + \mathcal{L}_{CE}(y', y)$
  $\mathcal{E}, \mathcal{D} = UpdateRule(\mathcal{E}, \mathcal{D}, \mathcal{L}')$
until Maximum number of iterations reached
return $\mathcal{E}, \mathcal{D}$

- Training can be carried out with alternate or simultaneous updates
  - We found both to perform similarly
- We make further use of labels to compute a standard classification loss
  - Found empirically to accelerate training
- No special scheme for selecting pairs

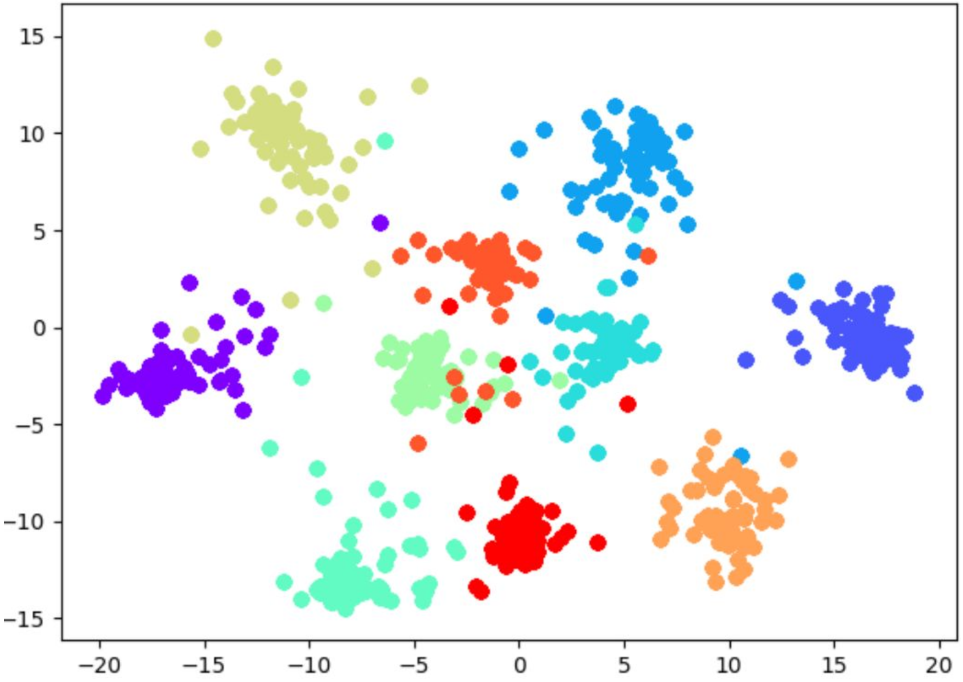
Outline

- Background
  - The verification problem
  - Distance metric learning / Metric learning

- Learning pseudo metric spaces
  - TL;DR
  - Method
  - Main results
  - Training details

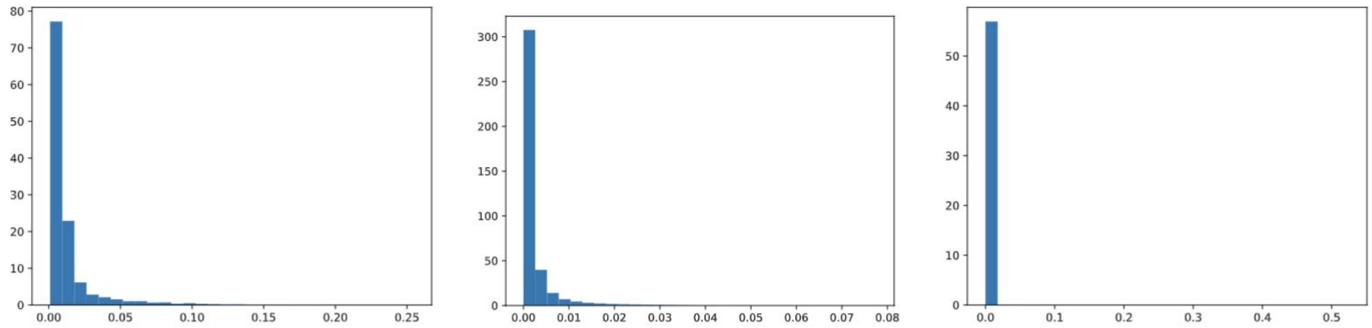
- **Evaluation**
  - Verifying standard distance properties in trained models
  - Proof-of-concept experiments on images
  - Open-set speaker verification

Properties of learned distance: embedding MNIST in $\mathbb{R}^2$

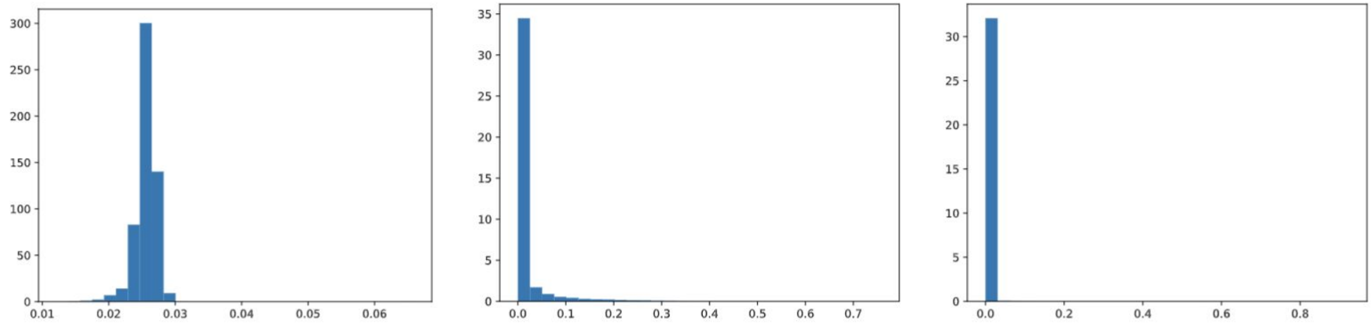


- Directly embedding pixels into $\mathbb{R}^2$
- Reasonably clustered test examples even if that was never enforced in the Euclidean sense

Verifying standard distance properties in trained models



(a) Distance to itself - Cifar-10. (b) Symmetry - Cifar-10. (c) Triangle inequality - Cifar-10.



(d) Distance to itself - VoxCeleb. (e) Symmetry - VoxCeleb. (f) Triangle inequality - VoxCeleb.

Evaluation of properties given by outputs of $\mathcal{D}' = 1 - \mathcal{D}$.

Proof-of-concept experiments on images

- Baselines: Standard Euclidean metric-learning with online hard negative mining

- Evaluation: Trials created via pairing of all test examples
    - Cifar-10: closed set
    - Mini-ImageNet: open set

- Our models perform at least as well while requiring no special pair selection strategy or complicated loss

		Scoring	*EER*	*1-AUC*
Cifar-10	Triplet	Cosine	3.80%	0.98%
	Proposed	E2E	3.43%	0.60%
		Cosine	3.56%	1.03%
		Cosine + E2E	3.42%	0.80%
Mini-ImageNet (Validation)	Triplet	Cosine	28.91%	21.58%
	Proposed	E2E	28.64%	21.01%
		Cosine	30.66%	23.70%
		Cosine + E2E	28.49%	20.90%
Mini-ImageNet (Test)	Triplet	Cosine	29.68%	22.56%
	Proposed	E2E	29.26%	22.04%
		Cosine	32.97%	27.34%
		Cosine + E2E	29.32%	22.24%

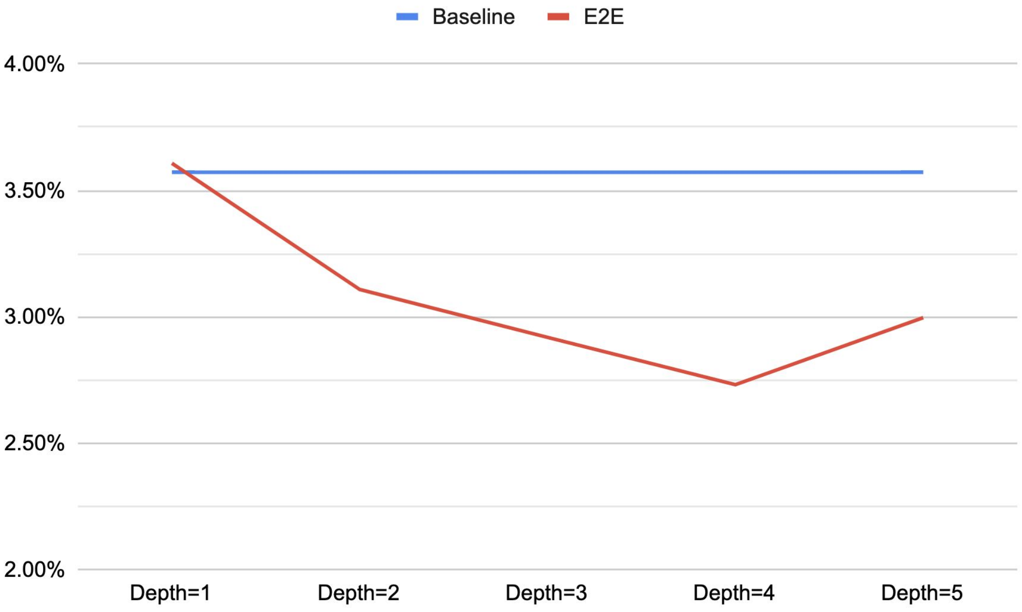
Large scale experiment on VoxCeleb

- Speaker verification on VoxCeleb:
  - Open-set: new speakers and languages at test time
- Able to outperform standard verification pipelines as well as recently introduced E2E approaches
- Ablation results indicate that the auxiliary loss boosts performance at no relevant cost
- More results in the paper for other partitions of the VoxCeleb test data

	Scoring	*Training set*	*EER*
VoxCeleb1 Test set			
Nagrani et al. (2017)	PLDA	VoxCeleb1	8.80%
Cai et al. (2018)	Cosine	VoxCeleb1	4.40%
Okabe et al. (2018)	Cosine	VoxCeleb1	3.85%
Hajibabaei & Dai (2018)	Cosine	VoxCeleb1	4.30%
Ravanelli & Bengio (2019)	Cosine	VoxCeleb1	5.80%
Chung et al. (2018)	Cosine	VoxCeleb2	3.95%
Xie et al. (2019)	Cosine	VoxCeleb2	3.22%
Hajavi & Etemad (2019)	Cosine	VoxCeleb2	4.26%
Xiang et al. (2019)	Cosine	VoxCeleb2	2.69%
Kaldi recipe[5]	PLDA	VoxCeleb2	2.51%
Proposed	Cosine	VoxCeleb2	4.97%
Proposed	E2E	VoxCeleb2	2.51%
Proposed	Cosine + E2E	VoxCeleb2	2.51%
Proposed	PLDA	VoxCeleb2	3.75%
Ablation $(-\mathcal{L}_{CE})$	E2E	VoxCeleb2	3.44%

Varying the depth of the distance model - ImageNet

- Distance models of increasing depth
- Baselines: Standard Euclidean metric-learning with online hard negative mining
- Evaluation: Trials created via pairing of all test examples
    - ImageNet: closed set
- Stable with respect to some of the introduced hyperparameters
    - Introduced hyperparameters can be easily tuned



Future directions

- Learn kernel functions for various tasks
- Learn space partitions in the pseudo metric spaces: prototypical nets style
- Borrow results from domain adaptation literature to derive generalization guarantees for the open-set case
    - Over pairs, new classes are simply new domains

Thank you!

joao.monteiro@emt.inrs.ca

https://github.com/joaomonteirof/e2e_verification