Interspeech 2024
1-5 September 2024, Kos, Greece

Predefined Prototypes for Intra-Class Separation and Disentanglement

Antonio Almudévar[1], *Théo Mariotte*[2,3], *Alfonso Ortega*[1], *Marie Tahon*[2], *Luis Vicente*[1], *Antonio Miguel*[1], *Eduardo Lleida*[1]

[1]ViVoLab, Aragón Institute for Engineering Research (I3A), University of Zaragoza, Spain
[2]LIUM, Le Mans University, Le Mans France
[3]LTCI, Télécom Paris, Institut Polytechnique de Paris, Palaiseau, France

almudevar@unizar.es

Abstract

Prototypical Learning is based on the idea that there is a point (which we call prototype) around which the embeddings of a class are clustered. It has shown promising results in scenarios with little labeled data or to design explainable models. Typically, prototypes are either defined as the average of the embeddings of a class or are designed to be trainable. In this work, we propose to predefine prototypes following human-specified criteria, which simplify the training pipeline and brings different advantages. Specifically, in this work we explore two of these advantages: increasing the inter-class separability of embeddings and disentangling embeddings with respect to different variance factors, which can translate into the possibility of having explainable predictions. Finally, we propose different experiments that help to understand our proposal and demonstrate empirically the mentioned advantages.

Index Terms: prototypical learning, predefined prototypes, intra-class separability, disentanglement, explainability

1. Introduction

Prototype theory is a concept from cognitive science and linguistics that suggests that humans categorize objects and ideas by comparing them to a mental representation of a prototype of each category or class [1]. These prototypes encapsulate the most central and representative features of each class. On the other hand, many machine learning systems base their operation on extracting representations [2, 3] (from now on we will refer to them interchangeably as embeddings), which are typically vectors that somehow encode the most representative features of the inputs. An approach that has gained popularity due to its good performance in a multitude of applications consists of introducing the idea of prototyping in machine learning systems [4]. These systems allow, in addition to extracting a representation of each input, to obtain a representation with the most representative features of the inputs of a class, which is called prototype. Moreover, these systems are typically trained so that all representations of the same class are close together in space. Therefore, as a result of these approximations, we end up having several sets of representations, each one in an area of space and corresponding to a class. Each of these sets of representations is around the prototype of that class. However, there are two properties that in general are not de facto fulfilled in this type of systems and that may be desirable for different tasks and applications. These desirable properties are:

- The embeddings of different classes are separated in space. This better use of all available space and usually results in a better performance in tasks such as classification, anomaly detection, object detection or biometric recognition [5–7].
- It is possible to associate some concrete dimensions of these representations with concrete human-understandable features so that a change of a feature produces changes in only a few dimensions of the space. This is has some advantages such as (i) having more control over data creation in generative models [8], or (ii) providing the ability to explain and interpret model predictions [9].

In this paper we propose a modification on the prototypical systems that preserves their default advantages and, in addition, allows solving the two problems presented. This modification consists in having the human predefine the prototypes before the training of the system, so that one of the objectives is that all the representations of a class are in an area of the space that has been decided with a human criterion and therefore can be explained. Typically, prototypes are trainable or simply computed as the average of the representations of a class and, to the best of our knowledge, the approach of imposing concrete conditions on them prior to training is an approach not explored in the literature. This may have several advantages, but in this paper we focus on its ability to achieve the two properties explained in the previous paragraph and whose are immediate to understand. First, if we predefine the prototypes before training, we can impose that they are far apart, by defining them as an orthogonal set of vectors, for example. On the other hand, we can define the prototypes so that some of their dimensions correspond to concrete human-interpretable factors.

To demonstrate with examples the two previous points, we propose two types of experiments demonstrating with them the correct operation of our proposal. In the first one, we solve a typical audio classification problem where we impose that representations of different classes are orthogonal to each other, showing that an implication of this is a better accuracy. On the other hand, we solve an emotion classification task where concrete dimensions of the representations correspond to acoustic parameters, allowing an explanation of how these parameters relate to the different emotions to be classified. More details on these experiments can be found at `https://github.com/antonioalmudevar/predefined_prototypes`

2. Related Work

Prototypical Learning. Its use was initially proposed for few-shot classification and class prototypes were calculated as the average of the few embeddings available for each class. During training, it is imposed that all embeddings are close to the prototype of their class, which implies that they are all close in space. Subsequently, different applications and variations of the prototypical systems have been proposed. For example, they have been used for unsupervised domain adaptation [10] or to build explainable systems [11, 12].

10.21437/Interspeech.2024-825

Intra-Class Separation. Due to the reasons explained in section1, different loss functions have been proposed that aim to separate embeddings of different classes in space. Among these loss functions, the following stand out: Center Loss [5], Focal Loss [6], Orthogonal Projection Loss [7] or that of the Variational Classifier [13]. The proposal of the present work allows defining the prototypes in such a way that they are far from each other and, consequently, the embeddings of the different classes are also far from each other.

Disentanglement. Although there is no consensus on its meaning, what is widely accepted is the intuition that it is the separation of the representations in the different variation factors of the data [2, 14]. That is, changing a variation factor in the data should change only a part of the representation. In [15] it is shown that it is not possible to achieve unsupervised disentanglement without inductive biases in the data or in the model. Therefore, multiple works have also been proposed focused on supervised disentanglement [16–18].

3. Proposed Method

3.1. General System Description

Let $\mathcal{D} = \{\{x^{(i)}, y^{(i)}, \alpha^{(i)}\}\}_{i=1}^{n}$ be a dataset where $x^{(i)} \in \mathbb{R}^p$ are each of the inputs, $y^{(i)} \in \mathbb{R}^C$ is the vector containing the class to which $x^{(i)}$ belongs and $\alpha^{(i)} \in \mathcal{A}$ is an abstract containing the factors of interest over $x^{(i)}$. We note that $\alpha^{(i)}$ can be extracted by an additional system and are the factors over which we intend to disentangle our representations. Deep Learning classifier systems are usually divided into two parts: $F_\theta : \mathbb{R}^p \rightarrow \mathbb{R}^k$, which we call embeddings extractor, and $G_\phi : \mathbb{R}^k \rightarrow \mathbb{R}^C$, which is the classifier network and is typically a linear layer plus a Softmax. On the other hand, in our system, we have $P : \mathbb{R}^C \times \mathcal{A} \rightarrow \mathbb{R}^k$, which we call the prototype extractor. We note that P does not depend on any trainable parameter, that is, it is not modified throughout the training, which is the main novelty of our work. From these three components we can compute embeddings as $z^{(i)} = F_\theta(x^{(i)})$, predictions as $\tilde{y}^{(i)} = G_\phi(z^{(i)})$ and prototypes as $p^{(i)} = P(y^{(i)}, \alpha^{(i)})$. The two objectives of this system are: (i) that $y^{(i)}$ and $\tilde{y}^{(i)}$ are similar to have a good classification performance, and (ii) that $z^{(i)}$ and $p^{(i)}$ are similar for the embedding extractor F_θ to behave similarly to P. This results in the following loss function:

$$\mathcal{L} = CE\left(y^{(i)}, \tilde{y}^{(i)}\right) + \lambda_p ||z^{(i)} - p^{(i)}||_2^2 \qquad (1)$$

where CE is the cross-entropy loss and λ_p is a hyperparameter that we set to $1/k$ in the experiments. In algorithm 1 we summarize the training procedure.

Finally, we propose that P is a multilinear map, i.e.:

- $P(ay^{(i_a)} + by^{(i_b)}, \alpha^{(i)}) = aP(y^{(i_a)}, \alpha^{(i)}) + bP(y^{(i_b)}, \alpha^{(i)})$, which allows working with soft labels (which implies, among other things, the ability to use regularization techniques such as mixup [19], widely used in audio classification).
- $P(y^{(i)}, a\alpha^{(i_a)} + b\alpha^{(i_b)}) = aP(y^{(i)}, \alpha^{(i_a)}) + bP(y^{(i)}, \alpha^{(i_b)})$, which provides ease in managing continuous variation factors.

3.2. Examples of Prototypes Extractors

The main novelty of our system with respect to others is the proposal that P is not modified during training and can also be defined by the human. This, despite being an unexplored idea to the best of our knowledge, may have a large number of applications. Specifically, in this paper we give two formulas that allow us to solve two very popular problems. These are: (i) maximize the distance between embeddings of different types and (ii) disentangle subsets of embeddings variables with respect to specified factors of variation. In the following, we give the formulas we propose to define P that allow to solve these problems.

Algorithm 1 Training Algorithm for the Predefined Prototypes System

Require: Dataset $\mathcal{D}$, Prototypes Extractor P, λ_p
  $\boldsymbol{\theta}, \boldsymbol{\phi} \leftarrow$ Initialize parameters
  repeat
    $\mathcal{D}^N \leftarrow \{(x^{(i)}, y^{(i)}, \alpha^{(i)})\}_{i=1}^{N}$ (Minibatch from $\mathcal{D}$)
    for $i = 1$ to N **do**
      $z^{(i)} \leftarrow F_\theta(x^{(i)})$
      $\tilde{y}^{(i)} \leftarrow G_\phi(z^{(i)})$
      $p^{(i)} \leftarrow P(y^{(i)}, \alpha^{(i)})$
    end for
    $\mathcal{L} \leftarrow \frac{1}{N}\sum_{i=1}^{N} CE\left(y^{(i)}, \tilde{y}^{(i)}\right) + \lambda_p ||z^{(i)} - p^{(i)}||_2^2$
    $\boldsymbol{\theta}, \boldsymbol{\phi} \leftarrow$ Update using gradients of $\mathcal{L}$
  until convergence of $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$

3.2.1. Maximize Intra-Class Separability

In this first example, we have that $P(y^{(i)}, \alpha^{(i)}) = P(y^{(i)}) : \mathbb{R}^C \rightarrow \mathbb{R}^k$, since our only goal is to separate embeddings of different classes independently of the input factors. Subsequently, we define (using Singular Value Decomposition, for example) an orthogonal basis $V = \{v^{(j)}\}_{j=1}^{C}$ such that $v^{(j)} \in \mathbb{R}^k$, so that $P(j) = v^{(j)}$. In this way, the prototypes of the different classes are orthogonal to each other and, consequently, the embeddings of elements of different classes will tend to be quasi-orthogonal, thus being far apart in space. Although typically the dimension of the embedding is higher than the number of classes, i.e., $k \geq C$, there may be scenarios in which this is not the case, which would prevent one from being able to define an orthogonal V basis as described. In the case where $k < C$, we propose to define an orthogonal basis $W = \{w^{(j)}\}_{j=1}^{C}$ such that $w^{(j)} \in \mathbb{R}^C$, subsequently define a Johnson Lindenstrauss Transform (JLT) $T : \mathbb{R}^C \rightarrow \mathbb{R}^k$, so that finally $v^{(j)} = T(w^{(j)}) \in \mathbb{R}^k$ for $j = 1, 2, \ldots, C$. JLTs are nearly distance-preserving transformations, so embeddings of different classes will end up being maximally separated in space [20]. Fast algorithms exist to compute these JLTs [21].

3.2.2. Embeddings Disentanglement with respect to Features

In this second example we define our prototype extractor as $P(y^{(i)}, \alpha^{(i)}) = P_\alpha(\alpha^{(i)}) : \mathcal{A} \rightarrow \mathbb{R}^k$, i.e., it depends on the factors but not on the labels. The way in which this $P_\alpha(\alpha^{(i)})$ is defined is highly dependent on the scenario and application. However, one way we have found to obtain a good classification performance and disentangled embeddings is to set $P_\alpha(\alpha^{(i)}) = \left(P'_\alpha(\alpha^{(i)}), \mathbf{0}\right)$, where $(\cdot, \cdot)$ means concatenation, $P'_\alpha(\alpha^{(i)}) : \mathcal{A} \rightarrow \mathbb{R}^{k_f}$ and $\mathbf{0}$ is the all-zero vector of length $k - k_f$. Thus, the goal is that we can clearly separate our embeddings into two parts: (i) one that depends on the factors and is disentangled with respect to them, and (ii) one whose value does not depend directly on the factors and whose elements are close to zero. The purpose of this second part of the embedding

is to give the model a greater degree of freedom in organizing its embeddings and to capture all the factors of variation that influence the predictions $\tilde{y}^{(i)}$ but are not captured in $\alpha^{(i)}$. Without this second part (or, equivalently, if $k = k_f$), predictions would be made from only $\alpha^{(i)}$, which would generally result in lower accuracy, since there are more factors that can affect $\tilde{y}^{(i)}$ and are not in $\alpha^{(i)}$. In fact, by comparing the two parts of the embeddings, we can obtain information the proportion the predictions determined by the known factors $\alpha^{(i)}$ and by unknown factors. Although one might think that defining the second part of the prototypes as $\mathbf{0}$ might lead to its elements tending to always be worth 0, this is not necessarily the case. As is the case in the Variational Autoencoder [22], the loss function to be optimized defined in the equation (1), is defined by two terms. The fact that cross-entropy must be minimized will lead our embeddings to differ across classes if this helps to minimize cross-entropy. If, on the other hand, our predictions can be made very accurately only from $\alpha^{(i)}$ (which is highly unlikely), the elements of the second part of the embeddings will tend to take values close to 0, which is positive for two reasons: (i) it indicates that there are no factors other than those in $\alpha^{(i)}$ that affect the predictions, which gives us new knowledge about our dataset; and (ii) it tells us that there are elements of the embeddings that are unnecessary, thus reducing the size of the embeddings and, consequently, the complexity of the model. The way to define P'_α is very scenario-dependent, so we do not give a general formula to define it. In section 4.3.2 we describe an emotion recognizer in which embeddings are disentangled with respect to a set of acoustic parameters and follow the previous formulation.

4. Experiments

4.1. Datasets

Environmental Sound Classification (ESC-50) [23] includes 2,000 ambient sound recordings categorized into 5 classes, each lasting 5 seconds. Throughout our experiments, we adhere to the standard 5-fold cross-validation approach.

Speech Commands V2 (KS2) [24] comprises 105,829 one-second clips of spoken keywords, each annotated with one of 35 word classes. Officially, it is segmented into 84,843 training clips, 9,981 test clips, and 11,005 validation clips.

IEMOCAP (ER) [25] consists of approximately 12 hours of speech showcasing four distinct emotions. Our evaluation uses the standard 5-fold cross-validation method proposed in [26].

4.2. Embeddings Extractors

ECAPA-TDNN [27] integrates attention mechanisms and parallel processing for efficient sequential data analysis, particularly suited for tasks like speech recognition.

Audio Spectrogram Transformer (AST) [28] renowned as the pioneer in using Transformer architectures for audio, has set a benchmark due to its exceptional performance. We initialize the model with pre-trained weights from Imagenet [29] and Audioset [30] and fine-tune it for each specific scenario.

BEATs [31] is an audio pre-training framework for learning representations from Audio Transformers, in which an acoustic tokenizer and a self-supervised audio model are optimized. We also start from the pre-trained model with Audioset.

4.3. Results

In all the next experiments, we use a sampling frequency of 16kHz. The inputs to every system are 128 mel-filterbank calculated in 25 ms windows every 10 ms. We normalize the mean and standard deviation to 0 and 0.5, respectively.

4.3.1. Audio Classification with Intra-Class Separation

To compare the performance of our proposal with others in the literature, we evaluate them on ESC-50 and KS2 using AST and BEATs as embedding extractors. The hyperparameters used to obtain the inputs and to train each of them are based on [28,31]. To evaluate our proposal, we compare it with identical systems where the only differentiating factor is the loss function. Specifically, we compare our loss function with cross-entropy, Focal Loss and OPL. The latter two, like our proposal, aim at increasing the distance between embeddings of different classes. They give excellent performances in different classification scenarios. We note that we have not been able to train them for KS2, since mixup is used, which results in having soft labels, and these two proposals do not allow working with soft labels. Conversely, our proposal does, since the prototype extractor is a multilinear map. In our proposal, we define the prototypes according to the procedure explained in 3.2.1. We have run all the described experiments three times and we show the mean and standard deviation of the accuracy in table 1. We see that in all cases except one, our system outperforms the rest on average. This seems to indicate that it is effective in separating embeddings of different classes and that this translates into better accuracy.

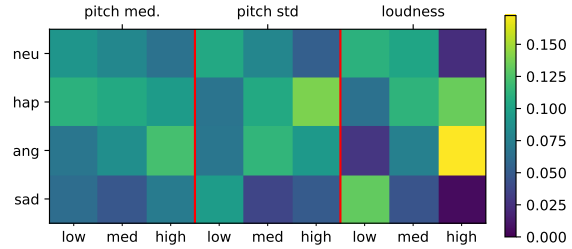


Figure 1: *Joint probabilities of each emotion and acoustic parameter (by levels) in the training dataset.*

4.3.2. Disentangled Emotion Recognition

Next we give an illustrative example of how to design a classifier whose embeddings are disentangled with respect to given factors of variation. We propose to use IEMOCAP as dataset to classify the emotions {*neutral, happy, angry, sad*} and choose {*pitch median, pitch std, loudness*} as factors with respect to which to disentangle our embeddings. We select only female voices, to minimize pitch differences between speakers. The reason for choosing these three acoustic parameters as factors is that they have been found to be related to emotions [32–34]. To extract *pitch median* and *pitch std*, we use [35]; and to obtain *loudness* we follow [36]. On the other hand, although P'_α can take continuous values as inputs and can be any type of function, here we discretize the values of the factors to three levels according to their value with respect to the $1/3$ and $2/3$ quantiles calculated in the training set. This facilitates interpretability so that, from now on, we will say that each factor is either *low*, *medium* or *high*. The joint probability of each emotion and level of these factors is shown in Figure 1. Subsequently, we define a coding function $C(\alpha_k^{(i)})$ as:

$$C(\alpha_k^{(i)}) = \begin{cases} (1,0,0) & \text{if } \alpha_k^{(i)} \text{ is } \textit{low} \\ (0,1,0) & \text{if } \alpha_k^{(i)} \text{ is } \textit{medium} \\ (0,0,1) & \text{if } \alpha_k^{(i)} \text{ is } \textit{high} \end{cases} \tag{2}$$

Table 1: *Accuracy for the different Datasets, Embeddings Extractors and Loss Functions*

	ESC-50		KS2	
	AST	BEATs	AST	BEATs
Cross-entropy	93.97 ± 0.21	91.05 ± 0.41	$\mathbf{92.05 \pm 0.04}$	88.94 ± 0.13
Focal Loss	94.40 ± 0.36	91.10 ± 0.49	-	-
OPL	94.11 ± 0.37	91.50 ± 0.20	-	-
Predefined Prototypes	$\mathbf{94.52 \pm 0.02}$	$\mathbf{91.72 \pm 0.30}$	91.45 ± 0.06	$\mathbf{89.42 \pm 0.09}$

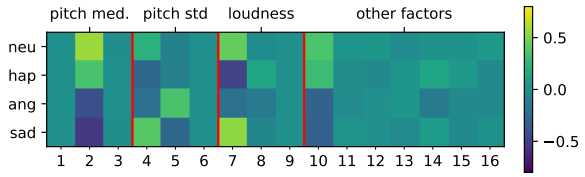
where $\alpha_1^{(i)}$, $\alpha_2^{(i)}$ and $\alpha_3^{(i)}$ are the *pitch median*, *pitch std* and *loudness* of the input $x^{(i)}$, respectively and $\alpha^{(i)} = \left\{\alpha_1^{(i)}, \alpha_2^{(i)}, \alpha_3^{(i)}\right\}$. Finally, we define $P'_\alpha(\alpha^{(i)}) = \left(C\left(\alpha_1^{(i)}\right), C\left(\alpha_2^{(i)}\right), C\left(\alpha_3^{(i)}\right)\right)$. Thus, an audio with *low pitch median*, *medium pitch std* and *high loudness* would correspond to a prototype $((1,0,0,0,1,0,0,0,1), \mathbf{0})$. As embedding extractor F_θ we use an ECAPA-TDNN with embedding size $k = 16$ (thus, length of $\mathbf{0}$ is 7). As classification net G_ϕ we use a linear layer without bias, i.e. $G_\phi \equiv W_\phi = (w_{jc}^\phi) \in \mathbb{R}^{k \times C}$ and the prediction is calculated as $\tilde{y}^{(i)} = W_\phi^T \cdot z^{(i)}$. Equivalently, we define the relevance matrix as:

$$\Gamma^{(i)} = (\gamma_{jc}^{(i)}) \text{ such that } \gamma_{jc}^{(i)} = w_{jc}^\phi \cdot z_j^{(i)} \tag{3}$$

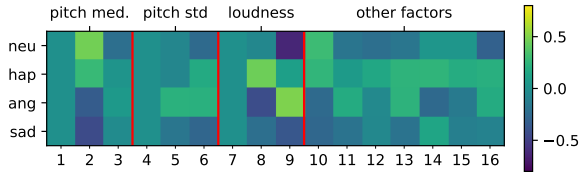
where $\gamma_{jc}^{(i)}$ contains information on how the j component of embedding $z^{(i)}$ influences the probability of class c. Next we analyze these relevance matrices of different embeddings shown in Figure 2 to illustrate this better.

- In Figure 2a we see a case where neutral emotion has been predicted with a probability of 0.68. We can see that having low *pitch std* and *loudness* added probabilities to both *sad* and *neutral*, but the fact that the *pitch median* is medium makes the model to decrease the probability of *sad*. As we can see in Figure 1, it is uncommon to have medium *pitch median* given the emotion *sad*.
- In Figure 2b we can suspect that the *loudness* is between *medium* and *high*, since the activated part of *high* adds probability to *angry*, but the activated part of *medium* subtracts to *angry* and adds to *happy*, which is the emotion with the highest probability. As we can see, the probability of having a *high loudness* when the emotion is *angry* is very high.
- In Figure 2c we see clearly that having a *high loudness* and *medium pitch std* has a great influence in predicting *angry*.
- In Figure 2d we can see that having *low pitch std* and *low loudness* increases the probability of *neutral* and *sad*. However, component 10 has a great influence on ultimately increasing the probability of *sad*.

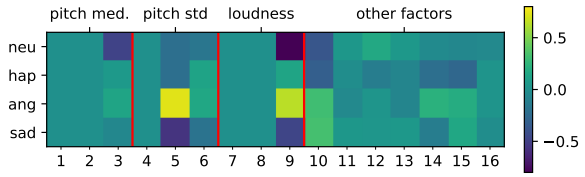
It is interesting to analyze component 10, since it seems to have similar behavior for *neutral* and *happy* and for *angry* and *sad* in all figures. This raises the suspicion that this component holds a factor related to the positivity or negativity of the emotion, but which we have not tried to disentangle explicitly. This is an example of the importance of giving freedom to a part of the embeddings. This part ends up learning factors that we have not explicitly tried to disentangle but seem to have influence on predictions and interpretable meaning. In addition, we see how the components associated with *other factors* are far from being 0 in most cases, as explained in section 3.2.2. Finally, we mention that the accuracy of the described system is 55.53 and that of one with the same characteristics but with cross-entropy loss 54.86, i.e. we obtain a similar performance in both cases.



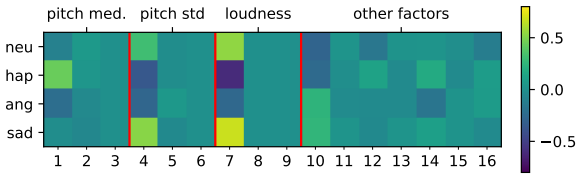
(a) $p_{neu} = 0.68$, $p_{hap} = 0.15$, $p_{ang} = 0.05$, $p_{sad} = 0.12$



(b) $p_{neu} = 0.02$, $p_{hap} = 0.87$, $p_{ang} = 0.10$, $p_{sad} = 0.01$



(c) $p_{neu} = 0.01$, $p_{hap} = 0.04$, $p_{ang} = 0.91$, $p_{sad} = 0.04$



(d) $p_{neu} = 0.20$, $p_{hap} = 0.10$, $p_{ang} = 0.08$, $p_{sad} = 0.62$

Figure 2: *Relevance matrices $\Gamma^{(i)}$ for different embeddings $z^{(i)}$ and their corresponding predictions $\tilde{y}^{(i)}$*

5. Conclusions

In this paper, we have proposed the possibility of defining the prototypes of a system before training and with a human criterion. We have argued some of the advantages and applications that this may have. Specifically, we have argued how this idea can be used to (i) increase the distance between embeddings of different classes and, consequently, improve the accuracy of a classifier; and (ii) disentangle embeddings with respect to given variation factors, being able to have more control over them and providing the possibility to explain predictions. However, the ultimate goal of the paper is to give the idea that it may be convenient in some cases to let the human define the prototypes instead of letting the system learn them by itself. The two proposed applications are intended to illustrate use cases in which applying this idea is both convenient and advantageous.

6. Acknowledgements

This work was supported by MCIN/AEI/10.13039/501100011033 under Grant PID2021-126061OB-C44, and in part by the Government of Aragón (Grant Group T36 23R). This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 101007666.

7. References

[1] E. H. Rosch, "Natural categories," *Cognitive psychology*, vol. 4, no. 3, pp. 328–350, 1973.

[2] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[3] A. Achille and S. Soatto, "Emergence of invariance and disentanglement in deep representations," *The Journal of Machine Learning Research*, vol. 19, no. 1, pp. 1947–1980, 2018.

[4] J. Klys, J. Snell, and R. Zemel, "Learning latent subspaces in variational autoencoders," *Advances in neural information processing systems*, vol. 31, 2018.

[5] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VII 14*. Springer, 2016, pp. 499–515.

[6] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[7] K. Ranasinghe, M. Naseer, M. Hayat, S. Khan, and F. S. Khan, "Orthogonal projection loss," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 12 333–12 343.

[8] I. Higgins, L. Matthey, A. Pal, C. P. Burgess, X. Glorot, M. M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework." *ICLR (Poster)*, vol. 3, 2017.

[9] X. Zhu, C. Xu, and D. Tao, "Where and what? examining interpretable disentangled representations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 5861–5870.

[10] Y. Pan, T. Yao, Y. Li, Y. Wang, C.-W. Ngo, and T. Mei, "Transferrable prototypical networks for unsupervised domain adaptation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 2239–2247.

[11] P. Zinemanas, M. Rocamora, M. Miron, F. Font, and X. Serra, "An interpretable deep learning model for automatic sound classification," *Electronics*, vol. 10, no. 7, p. 850, 2021.

[12] T. Mariotte, A. Almudévar, M. Tahon, and A. Ortega, "An explainable proxy model for multiabel audio segmentation," 2024.

[13] A. Almudévar, A. Ortega, L. Vicente, A. Miguel, and E. Lleida, "Variational Classifier for Unsupervised Anomalous Sound Detection under Domain Generalization," in *Proc. INTERSPEECH 2023*, 2023, pp. 2823–2827.

[14] I. Higgins, D. Amos, D. Pfau, S. Racaniere, L. Matthey, D. Rezende, and A. Lerchner, "Towards a definition of disentangled representations," *arXiv preprint arXiv:1812.02230*, 2018.

[15] F. Locatello, S. Bauer, M. Lucic, G. Raetsch, S. Gelly, B. Schölkopf, and O. Bachem, "Challenging common assumptions in the unsupervised learning of disentangled representations," in *international conference on machine learning*. PMLR, 2019, pp. 4114–4124.

[16] F. Locatello, B. Poole, G. Rätsch, B. Schölkopf, O. Bachem, and M. Tschannen, "Weakly-supervised disentanglement without compromises," in *International conference on machine learning*. PMLR, 2020, pp. 6348–6359.

[17] B. Estermann and R. Wattenhofer, "Dava: Disentangling adversarial variational autoencoder," *arXiv preprint arXiv:2303.01384*, 2023.

[18] A. Almudévar, T. Mariotte, A. Ortega, and M. Tahon, "Unsupervised multiple domain translation through controlled disentanglement in variational autoencoder," 2024.

[19] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[20] W. B. Johnson, J. Lindenstrauss, and G. Schechtman, "Extensions of lipschitz maps into banach spaces," *Israel Journal of Mathematics*, vol. 54, no. 2, pp. 129–138, 1986.

[21] N. Ailon and B. Chazelle, "Approximate nearest neighbors and the fast johnson-lindenstrauss transform," in *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, 2006, pp. 557–563.

[22] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[23] K. J. Piczak, "Esc: Dataset for environmental sound classification," in *Proceedings of the 23rd ACM international conference on Multimedia*, 2015, pp. 1015–1018.

[24] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *arXiv preprint arXiv:1804.03209*, 2018.

[25] C. Busso, M. Bulut, C.-C. Lee, A. Kazemzadeh, E. Mower, S. Kim, J. N. Chang, S. Lee, and S. S. Narayanan, "Iemocap: Interactive emotional dyadic motion capture database," *Language resources and evaluation*, vol. 42, pp. 335–359, 2008.

[26] S.-w. Yang, P.-H. Chi, Y.-S. Chuang, C.-I. J. Lai, K. Lakhotia, Y. Y. Lin, A. T. Liu, J. Shi, X. Chang, G.-T. Lin *et al.*, "Superb: Speech processing universal performance benchmark," *arXiv preprint arXiv:2105.01051*, 2021.

[27] B. Desplanques, J. Thienpondt, and K. Demuynck, "Ecapa-tdnn: Emphasized channel attention, propagation and aggregation in tdnn based speaker verification," *arXiv preprint arXiv:2005.07143*, 2020.

[28] Y. Gong, Y.-A. Chung, and J. Glass, "Ast: Audio spectrogram transformer," *arXiv preprint arXiv:2104.01778*, 2021.

[29] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[30] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2017, pp. 776–780.

[31] S. Chen, Y. Wu, C. Wang, S. Liu, D. Tompkins, Z. Chen, and F. Wei, "Beats: Audio pre-training with acoustic tokenizers," *arXiv preprint arXiv:2212.09058*, 2022.

[32] I. R. Murray and J. L. Arnott, "Toward the simulation of emotion in synthetic speech: A review of the literature on human vocal emotion," *The Journal of the Acoustical Society of America*, vol. 93, no. 2, pp. 1097–1108, 1993.

[33] K. Hammerschmidt and U. Jürgens, "Acoustical correlates of affective prosody," *Journal of voice*, vol. 21, no. 5, pp. 531–540, 2007.

[34] D. A. Sauter, F. Eisner, A. J. Calder, and S. K. Scott, "Perceptual cues in nonverbal vocal expressions of emotion," *Quarterly journal of experimental psychology*, vol. 63, no. 11, pp. 2251–2272, 2010.

[35] D. Talkin and W. B. Kleijn, "A robust algorithm for pitch tracking (rapt)," *Speech coding and synthesis*, vol. 495, p. 518, 1995.

[36] I. Recommendation, "Itu-r bs. 1770-4," *Algorithms to measure audio programme loudness and true-peak audio level*, 2015.