UNIVERSIDADE DE SÃO PAULO

Instituto de Ciências Matemáticas e de Computação

Robust image features creation by learning how to merge visual and semantic attributes

Damarens Oliveira de Resende

Dissertação de Mestrado do Programa de Pós-Graduação em Ciências
de Computação e Matemática Computacional (PPG-CCMC)

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Damare Oliveira de Resende

Robust image features creation by learning how to merge visual and semantic attributes

Dissertation submitted to the Institute of Mathematics and Computer Sciences – ICMC-USP – in accordance with the requirements of the Computer and Mathematical Sciences Graduate Program, for the degree of Master in Science. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Prof. Dr. Moacir Antonelli Ponti

USP – São Carlos
March 2021

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

A634m Antonelli, Humberto Lidio
 Modelo de teses e dissertações em LaTeX do ICMC /
Humberto Lidio Antonelli; orientador Renata Pontin
de Mattos Fortes. -- São Carlos, 2018.
 81 p.

 Tese (Doutorado - Programa de Pós-Graduação em
Ciências de Computação e Matemática Computacional) --
Instituto de Ciências Matemáticas e de Computação,
Universidade de São Paulo, 2018.

 1. Modelo. 2. Monografia de qualificação. 3.
Dissertação. 4. Tese. 5. Latex. I. Fortes, Renata
Pontin de Mattos, orient. II. Título.

Damare Oliveira de Resende

**Criando características de imagens robustas por meio do
aprendizado da fusão de atributos visuais e semânticos**

Dissertação apresentada ao Instituto de Ciências
Matemáticas e de Computação – ICMC-USP,
como parte dos requisitos para obtenção do título
de Mestra em Ciências – Ciências de Computação e
Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e
Matemática Computacional

Orientador: Prof. Dr. Moacir Antonelli Ponti

USP – São Carlos
Março de 2021

*This work is dedicated to adult children who
are not afraid of shooting the moon dreaming of at least reaching the stars.
In particular, to researchers at the Institute of Mathematical and Computer Sciences (ICMC).*

ACKNOWLEDGEMENTS

The prominent acknowledgments are directed to my parents and brother that encourage me to fight for my dreams and be persistent, to my advisor that welcomed me into his research group, taught me, inspired and boosted me during the project, and to all those who contributed to making this work possible. In particular, to Luisa Anastácio and Priscila Kido Martins, friends who support and inspire me, and were by my side in the darkest days.

Special thanks to the research group for Visualization, Images, and Computer Graphics (VICG) and to the University of São Paulo (USP), who provided me with space and support to carry out this research, as well as to its members who shared knowledge so that my skills could be improved.

“Failure is simply the opportunity to begin again, this time more intelligently”
(Henry Ford)

RESUMO

RESENDE, D. O. **Criando características de imagens robustas por meio do aprendizado da fusão de atributos visuais e semânticos**. 2021. 71 p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2021.

Existem vantagens conhecidas em usar atributos semânticos para melhorar a representação de imagens. No entanto, o estudo de como esses atributos melhoram subespaços visuais e os efeitos sobre a classificação de dados grosseiros e granulares ainda estava para ser investigado. Esta pesquisa reporta um Codificador Visual-Semântico (VSE) construído a partir de um autoencoder sub completo, formado por uma rede neural que combina características semânticas e visuais para formar um espaço compacto que contém as propriedades mais relevantes de cada domínio. É observado empiricamente que o espaço latente aprendido pode melhor representar as características de imagens e inclusive permite a interpretação dos resultados baseado na natureza dos atributos semânticos, oferecendo um caminho para a aprendizagem explicável. Os experimentos foram realizados em quatro bases de dados *benchmark* onde o VSE foi comparado com algoritmos do estado-da-arte para a redução de dimensionalidade. O algoritmo se mostra robusto para até 20% de degradação dos dados semânticos e é tão eficiente quanto o LLE para aprender um espaço de baixa dimensionalidade com rica representatividade, oferecendo possibilidades para trabalhos futuros na aplicação de um coletor automático de dados semânticos para melhorar as representações. Ademais, o estudo sugere experimentalmente que a inclusão de conceitos de alto nível à representação de imagens adiciona linearidade ao espaço de características, permitindo que o PCA tenha boa performance na combinação de propriedades visuais e semânticas para melhorar a separabilidade das classes. Por fim, experimentos foram realizados no âmbito de *zero-shot learning*, onde VSE e PCA superam SAE, o algoritmo estado-da-arte proposto por [Kodirov, Xiang and Gong \(2017\)](#), e JDL, o *framework* de aprendizado discriminativo conjunto proposto por [Zhang and Saligrama \(2016\)](#), o que demonstra a viabilidade da mesclagem de dados semânticos e visuais nas etapas de treino e teste para aprender aspectos que transcendem as fronteiras de classes e permitem a classificação de dados desconhecidos.

Palavras-chave: classificação de imagens, aprendizado de características, aprendizado de variedades, autoencoder.

ABSTRACT

RESENDE, D. O. **Robust image features creation by learning how to merge visual and semantic attributes**. 2021. 71 p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2021.

There are known advantages of using semantic attributes to improve image representation. However, studying how to use such attributes to improve visual subspaces and its effects on coarse and fine-grained classification were still to be investigated. This research report a Visual-Semantic Encoder (VSE) built from a neural network undercomplete autoencoder, that combines visual features and semantic attributes to form a compact subspace containing each domain's most relevant properties. It is observed empirically that a learned latent space can better represent image features and even allow one to interpret results in the light of the nature of semantic attributes, offering a path for explainable learning. Experiments were performed in four benchmark datasets where VSE was compared against state-of-the-art algorithms for dimensionality reduction. The algorithm shows to be robust for up to 20% degradation of semantic attributes and is as efficient as LLE for learning a low-dimensional feature space with rich class representativeness, offering possibilities for future work on the deployment of an automatic gathering of semantic data to improve representations. Additionally, the study suggests experimentally that adding high-level concepts to image representations adds linearity to the feature space, allowing PCA to perform well in combining visual and semantic features for enhancing class separability. At last, experiments were performed for zero-shot learning, where VSE and PCA outperform SAE, the state-of-the-art algorithm proposed by [Kodirov, Xiang and Gong \(2017\)](#), and JDL, the joint discriminative learning framework proposed by [Zhang and Saligrama \(2016\)](#), which demonstrates the viability of merging semantic and visual data at both training and test time for learning aspects that transcend class boundaries that allow the classification of unseen data.

Keywords: image classification, feature learning, manifold learning, autoencoder.

LIST OF FIGURES

Figure 1 – Illustration of semantic features extraction and representation. Figure (a) shows binary attributes collected from text annotations. Figure (b) displays the extraction process of an object detection algorithm using continuous values.	26
Figure 2 – Example of a gray-scale image representation in pixels. The zoom in Lenna's eye show each component of the matrix o pixels that composes it. This image is represented in one channel only, hence only black and white levels can be shown. Picture adapted from Gonzales and Woods (2002).	28
Figure 3 – Representation of the feature extraction process in a deep neural network. The first layers retrieve properties such as corners and edges and last layers learn high level properties such as object parts, which are built from the hierarchy of features learned in the previous layers. Picture adapted from Goodfellow <i>et al.</i> (2016).	29
Figure 4 – Representation of the model developed by Kodirov, Xiang and Gong (2017). The authors use an autoencoder that encodes the visual features (X) in the semantic space (S), and decodes it back to the visual space. With this approach they were able to mitigate the project domain shift problem. Picture adapted from Kodirov, Xiang and Gong (2017).	39
Figure 5 – Representation of the model developed by Zhang and Saligrama (2016). The authors define the zero-shot learning problem as a binary problem, where the model aims to identify if the source and target domains examples belong to the same class or not. They do that by mapping theses classes in a dictionary. Picture adapted from Zhang and Saligrama (2016).	40
Figure 6 – The structure of VSE for an autoencoder with three layers in the encoder and three layers in the decoder. The number of neurons follow Equation 4.2. κ values are highlighted in red.	42
Figure 7 – Illustration of feature extraction, training, and classification processes. The upper left shows how visual features are acquired, and the lower left how the semantic descriptors are built. Both these data are normalized, concatenated, and then input to VSE. After training, the code generated is classified by a linear SVM. Note that classes are not used in the training process.	43

Figure 8 – Images from AwA2 on the first 4 columns (left to right) from classes: german shepherd, zebra, humpback whale and hamster, and from CUB on the last 4 columns from classes: sooty albatross, glaucous-winged gull, chestnut-sided warbler, and brewer blackbird. Note images from the same class may be captured from different perspectives, may have confusing objects, clutter and varying resolution.	48
Figure 9 – Images from SUN on the first 4 columns (left to right) from classes: dock, embassy, kitchen and park, and from aP&Y on the last 4 columns from classes: aeroplane, person, mug, and centaur. Note that some scenes may not share the same set of attributes, that images can contain other valid classes within it, such as the person holding a cat and a mug with the picture of a cow, or even a mixture of different classes such as a centaur (person/horse).	48
Figure 10 – PCA 2D plot of visual, semantic and VSE features. This image illustrates how separable a subset of classes of AwA2 is in each domain. The robustness of the space formed can also be observed with the Silhouette Score. We show terrestrial and aquatic animals, where terrestrial animals can be felines, equines or a dalmatian. The later was introduced as a confusion category. One can observe that the merged domain formed by VSE preserves the semantic and visual relations in space, while improving the cohesion and separability of the clusters.	51
Figure 11 – PCA 2D plot of VSE features split into different types: colors, texture, shape, and parts. Note that depending on the set of semantic attributes chosen, the arrangement of the clusters changes in the latent space, reflecting logical concepts such as black and white animals together, animals with spots and patches together, aquatic animals, and animals with four legs together.	53
Figure 12 – Training loss and classification accuracy along all epochs. Training loss is measured based on the Mean Square Error of the reconstruction of the autoencoder, while classification accuracy corresponds to the results of the <i>balanced accuracy score</i> applied over SVM predictions. In the illustration, <i>i_train</i> and <i>i_val</i> are the training and validation results when using Inception CNN to extract the visual features, while <i>r_train</i> and <i>r_val</i> are the same but for ResNet CNN.	57
Figure 13 – Results for SVM classification with perturbations in the semantic data at eleven different levels, being 0% classification with no degradation, and 100% classification with all values of the semantic data at test phase replaced by random numbers.	58

LIST OF TABLES

Table 1	– Resumo do número de imagens, categorias e arranjo dos atributos semânticos para as bases de dados AWA, CUB, SUN e aP&Y. Os valores com (*) não são os mesmos observados na literatura.	47
Table 2	– PCA variance ratio of the first ten components of the visual (VIS), semantic (SEM), and merged (VSE) spaces for AWA2 dataset, where a subset of twelve classes of animals was chosen: Persian cat, Siamese cat, tiger, leopard, killer whale, dolphin, blue whale, walrus, horse, giraffe, dalmatian, zebra. The first two components of the space formed are displayed in Figure 10.	52
Table 3	– PCA variance ratio of the first ten components of the semantic (SEM) and merged (VSE) domains for AWA2 dataset. In this case, different sets of semantic attributes were chosen. They represent color (COL), texture (TEX), shape (SHP), and parts (PRT) of twelve classes of animals: Persian cat, Siamese cat, tiger, leopard, killer whale, dolphin, blue whale, walrus, horse, giraffe, dalmatian, zebra. The first two components of the space formed are displayed in Figure 11. The attributes used in each category were listed in Section 5.1.1.1.	54
Table 4	– Results for SVM classification with 0% of semantic degradation. Accuracies are computed using the metric <i>balanced accuracy score</i> over 5 stratified folds in an 80 by 20 proportion for determining training and validation data for SAE, VSE, S2S, PCA, ISO, and LLE algorithms. SVM classification is performed over the code created by these algorithms at validation time.	56
Table 5	– Results for ZSL classification using $f_{zs} : V \rightarrow S$ approach (SAE, SAE^* and SEC), and $f_{zs} : V \oplus S \rightarrow K$ (VSE, JDL and PCA). SAE and JDL are shown separately because these are the results presented in the original papers, Kodirov, Xiang and Gong (2017) and Zhang and Saligrama (2016) respectively. For the former, only Inception V1 (SZEGEDY <i>et al.</i> , 2015) was used. For the latter, visual attributes were acquired from AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) and VGGNet 19 (SIMONYAN; ZISSERMAN, 2014). The other results were retrieved from the applications coded in this project.	59

CONTENTS

1	INTRODUCTION	21
2	THEORETICAL BACKGROUND	25
2.1	Feature Learning	25
2.1.1	<i>Semantic Features</i>	25
2.1.2	<i>Visual Features</i>	27
2.2	Deep learning	28
2.2.1	<i>Deep Neural Networks</i>	28
2.2.2	<i>Convolutional Neural Networks</i>	30
2.3	Unsupervised Learning	30
2.3.1	<i>Autoencoders</i>	31
2.3.2	<i>Principal Component Analysis (PCA)</i>	33
2.3.3	<i>Manifold Learning</i>	33
2.3.3.1	<i>Isomap</i>	33
2.3.3.2	<i>Locally Linear Embedding (LLE)</i>	34
2.4	Zero-Shot Learning	34
3	RELATED WORK	37
3.1	ZSL using an autoencoder	38
3.2	ZSL by merging visual and semantic data	39
4	METHOD	41
4.1	Model Formulation	41
4.2	Feature extraction	42
4.2.1	<i>Visual Features</i>	43
4.2.2	<i>Semantic Features</i>	44
4.3	Model Training	45
4.4	Datasets	46
4.5	Evaluation	49
4.5.1	<i>Space Separability</i>	49
4.5.2	<i>Linear Classification</i>	49
4.5.3	<i>Zero-Shot Learning</i>	50
5	RESULTS AND DISCUSSION	51

5.1	Results	51
5.1.1	<i>Space separability</i>	51
5.1.1.1	<i>Influence of different semantic features</i>	53
5.1.2	<i>Linear Classification</i>	55
5.1.2.1	<i>Training History Analysis</i>	56
5.1.2.2	<i>Semantic Degradation</i>	57
5.1.3	<i>Zero-Shot Learning</i>	57
5.2	Discussion	59
6	CONCLUSION	63
BIBLIOGRAPHY		65
GLOSSARY		71

INTRODUCTION

Pattern recognition consists of extracting characteristics and attributes in order to identify classes by discarding noise and irrelevant information. This process includes the capacity for classification and the capacity to recognize patterns that make the object being analyzed ineligible to other classes (GONZALEZ; THOMASON, 1978), where the characterization of objects is based on the definition of attributes that indicate particular aspects related to its physical and functional properties (SILBERER; FERRARI; LAPATA, 2013). Mostly, two recognition methodologies are used: the numerical one, which consists of the formal description of the process, the analysis of characteristics, the design of the classifier, and the analysis of the formed groups; and the syntactic or structural form, which deals with the description of complex patterns through the hierarchical composition of sub-patterns (BEZDEK *et al.*, 1999).

In the scope of image classification, a classical and important task in computer vision, Convolutional Neural Networks (CNNs) have shown to be robust models, capable of classifying hundreds of different categories (HE *et al.*, 2016; SZEGEDY *et al.*, 2015; DENG *et al.*, 2009). However, the number of distinct classes these models can discriminate increases with the amount of data they need for training them. When discriminating hundreds of different objects, as provided in the ImageNet Large-Scale Visual Recognition Challenge, for instance, these models need thousands or even millions of examples to distinguish the groups. Nonetheless, even with extensive data, it will never be enough compared to the complexity of the world (FARHADI *et al.*, 2009), considering that in order to make accurate inferences while being generic enough, a neural network needs to learn representations based on several examples of each class.

Humans, on the other hand, are capable of identifying and discriminating around 30 thousand different objects (BIEDERMAN, 1987), more than that, they are able to abstract concepts from the visual representation based on prior knowledge gained from their experience, and can recognize unseen objects when provided high-level descriptions (LAMPERT; NICKISCH; HARMELING, 2009; XU *et al.*, 2018). These properties are not restricted to categories and can express similarities among different concepts. Encouraged by this idea, studies employ semantic

attributes with deep networks to address challenging scenarios with clutter or pose variation in fine-grained data, and tasks such as zero-shot learning and multilabel classification. Nonetheless, how these attributes improve the visual subspaces and its effects on coarse and fine-grained classification were yet to be studied.

In this context, we propose and evaluate an undercomplete autoencoder that combines semantic data, formed by the mapping of attributes elucidated by humans to describe a particular class, and visual data, formed by the attributes extracted by machines from images, into one lower dimension representation, learning the underlying arrangement of the domains of both types of properties. In this work, we focus on the analysis of the visual and semantic feature spaces and in combining them to improve classification for both coarse, e.g. AWA1 and AWA2 datasets (XIAN *et al.*, 2018a) and aPascal&Yahoo (FARHADI *et al.*, 2009) database, and fine-grained, e.g. CUB200-2011 (WAH *et al.*, 2011) and SUN Attributes (PATTERSON; HAYS, 2012) datasets, aiming to enrich visual features obtained from pre-trained CNNs: GoogLeNet/Inception V1 (I1) (SZEGEDY *et al.*, 2015), Inception V3 (I3) (SZEGEDY *et al.*, 2016), and ResNet50 (RN) (HE *et al.*, 2016), and through that investigate the complementarity relationship between visual global aspects and the semantic attributes in the image domain.

With respect to our method, called Visual-Semantic Encoder (VSE), the encoder-decoder structure is responsible for choosing the most representative features of an object, reducing its dimensionality, but also allowing its reconstruction. According to Friedman (1997), learning methods that rely on nearest neighbors procedures, such as bagging and aggregating classifiers, suffer from the curse-of-dimensionality driven by the sparsity of the data, which makes similar groups harder to detect once data is too spread in the feature space. Consequently, the amount of data needed to obtain reliable classification results in such spaces grows exponentially with the dimensionality of the examples. Hence, when employing models with high-dimensionality in terms of parameters, one also has to worry about learning the generalization properties of machines (MELLO; PONTI, 2018).

We set the encoding to the same size as the semantic data available. This architecture was chosen for two reasons: (i) direct comparison to Kodirov, Xiang and Gong (2017); and (ii) according to Ranzato *et al.* (2007), Almousli and Vincent (2013), by making the code smaller than the input, we can reduce the reconstruction error while attempting to learning an innate dimension of the input data. We hypothesize that the merge of the visual and semantic domains in a lower-dimensional space will enhance class discrimination in the feature space, and that the embedded representation will generalize data without the need for additional examples, since the semantic data already provide inter and extra class properties, usually learnt through the analysis of hundreds of images of a particular group, mitigating the issue of the curse-of-dimensionality stated by Friedman (1997). In addition, managing a large number of features, some of which irrelevant or misleading, can burden the classification task, while a dimensionality reduction algorithm might underlie forces governing the database (CAYTON, 2005).

Here we: (i) show how a mixed visual-semantic latent space can better represent image features, (ii) visualize different semantic attributes and its influence to the formation of these representations to better interpret/explain the learned space; (iii) study the impact of using degraded semantic data, showing that our method is as robust and compact as competing methods, which can benefit real-world applications in which semantic attributes are automatically estimated; and (iv) evaluate the applicability of the combination of semantic and visual data to zero-shot learning applications, which aim to classify unseen classes at test level based on shared high level concepts learned at training level.

Our experiments indicate that VSE successfully merges features from semantic and visual domains in a logical manner, creating a latent space more distinguishable among all categories and robust to semantic data estimates. We show that in addition to better representing the features of classes for both coarse and fine-grained classification, VSE embeds semantic concepts without losing visual aspects extracted from images. Additionally, the proposed algorithm presents to be as robust as state-of-the-art methods for dimensionality reduction, such as PCA, Isomap, and Locally Linear Embedding, with the advantage of being able to reconstruct the original data. We also observe experimentally that the addition of semantic data to the representation introduces linearity in the feature space, enabling PCA to perform well on coarse datasets, while for fine-grained data, non-linear methods are better. Nevertheless, there are cases where the embedding process might not even be necessary, being the concatenation of both domains sufficient for improving the representation.

When it comes to zero-shot learning, VSE outperforms SAE, the state-of-the-art method proposed by [Kodirov, Xiang and Gong \(2017\)](#), and JDL, a joint discriminative learning framework proposed by [Zhang and Saligrama \(2016\)](#). Two approaches were considered to address this problem: 1) projection from visual space to the semantic space; 2) creating a new representation from a combination of visual and semantic features. In order to do that, we modify VSE's learning method to compare it directly to SAE and apply PCA for comparing zero-shot performance to VSE's original proposition, as well as JDL. We show that the use of combined data boosts the performance of classification in this application when the comparison of the embedding formed with the prototypes is direct. Therefore, depending on the availability of data, real-world applications can take advantage of this combination.

The rest of this text is organized as follows: chapter 2 presents some important concepts studied in this research; chapter 3 shows articles in the literature that work with the combination of semantic attributes to the visual data; chapter 4 presents in details the method developed; chapter 5 shows some results and discussions; and at last chapter 6 presents the conclusions.

THEORETICAL BACKGROUND

2.1 Feature Learning

Feature learning is based on the acquisition of knowledge through pattern recognition based on raw data (GOODFELLOW *et al.*, 2016). It is given by the capacity of making inferences about an input value based on the knowledge obtained from the analysis of several examples. It is an inductive learning, where logical inferences allow the model to make general conclusions about the dataset studied (MICHALSKI; CARBONELL; MITCHELL, 2013). This type of learning depends on the capacity of a computational model to extract attributes from objects, and with that, build a function f capable of mapping each example x in the desired output y , by describing, comparing, and categorizing them (FARHADI *et al.*, 2009; MELLO; PONTI, 2018).

In Computer Vision, feature learning is based on pattern recognition from the extraction of attributes from images and videos. These features represent different properties of the objects being described, and depending on the complexity of the computational model used, they belong to different levels of abstraction. The properties extracted are related to visual aspects but can hold semantic meaning. Details are provided in the next sessions.

2.1.1 Semantic Features

The prototype theory from cognitive psychology states that specific categories of an object can be defined by unique prototypes based on the description of attributes (ROSCH, 1975; ROSCH, 1999). For instance, fishes swim and have flippers while birds fly and have feathers. In addition, the human visual system often associates semantic and visual similarities to categorize an object (ROSCH, 1975; ROSCH, 1999). For example, mugs and cups are used for drinking, but only mugs have handles; benches and chairs have a seat and a back, but benches are wider.

These characteristics are called semantic attributes and report high-level aspects about a particular object. They can describe parts: “has horn”, “has tail”; shape: “cylindrical”, “spherical”;

materials: “metal”, “wood”; colors: “yellow”, “white”; textures: “spots”, “stripes”; or even functions: “drink”, “read” (FARHADI *et al.*, 2009; SU; JURIE, 2012; AKATA *et al.*, 2015; JAYARAMAN; GRAUMAN, 2014).

The semantic features can be retrieved from human annotations, what may be costly once thousands of examples have to be manually categorized and described; or automatically acquired from off-the-shelf models such as Region Based Convolutional Natural Networks (RCNNs) (GIRSHICK *et al.*, 2014), object detection networks (REDMON; FARHADI, 2018), or even text mining algorithms (MIKOLOV *et al.*, 2013a; MIKOLOV *et al.*, 2013b). In this context, semantic information has become widely available due to the wide use of social media and digital devices, which provides rich multimedia data with tags and comments that might characterize the content of an image (ZHANG *et al.*, 2018).

(a) Feature extraction based on annotations (b) Feature extraction based on object detection
Figure 1 – Illustration of semantic features extraction and representation. Figure (a) shows binary attributes collected from text annotations. Figure (b) displays the extraction process of an object detection algorithm using continuous values.

Figure 1 illustrates examples of semantic attributes. Figure 1 (a) shows a description based on annotated data. One can notice that despite the back of the leopard is not displayed, the description indicates the presence of four legs and a tail. This is because this prototype is based on the label of an image. Since all leopards have four legs and a tail, even though these features are not in the image, the semantic descriptor will have it. Figure 1 (b), on the other hand, shows an example of an image described by an object detection network. It places bounding boxes around each object and usually describes content of an image. In this case, the semantic prototype is unique for each image. For instance, kitchens usually have a stove. In this example,

this object was not detected but can be found in other images of the same class.

As Figure 1 shows, the semantic description can be represented by binary (a) or continuous values (b). However, according to Akata *et al.* (2015), Xian *et al.* (2016), continuous data better describes the semantic domain. Semantic attributes are often used in applications such as zero-shot learning (AKATA *et al.*, 2015; KODIROV; XIANG; GONG, 2017; ZHANG; SALIGRAMA, 2016), multi-label classification (ZHANG *et al.*, 2018; WANG *et al.*, 2016), image retrieval (JUAN *et al.*, 2019), image annotation (REN *et al.*, 2015) and scene description (PATTERSON; HAYS, 2012; PATTERSON *et al.*, 2014).

2.1.2 Visual Features

“An image is worth more than a thousand words”, Confucio. This Chinese philosopher’s sentence summarizes the importance of visual features to humans. According to Russ (1990), the visual system is the principal means of humanity in interacting with an environment, and about 60% of the sensory inputs to the brain come from it. Therefore, it is only natural to use computerized methods for image measurement.

The visual features are quantitative values related to size, shape, position, density, color, and texture, which can help a machine recognize, count, or measure objects within an image or video. These attributes are acquired based on the analysis of the distribution and arrangement of pixels in an image or frame, resulting from many transformations and computations in that input, and involves the selection of the parameters that are more important for the recognition of an object (RUSS, 1990; GONZALES; WOODS, 2002).

In a computer, an image is represented by a multidimensional array containing $m \times n \times c$ values, where m is the number of rows, n the number of columns, and c the number of channels, which in case of RGB inputs is equal to 3. Each component is called a pixel, where $p \in \mathbb{N}^*$. Depending on the bitmap, the color range may vary in representation. Pixels in 8-bit images, for instance, can have 256 different insensitive levels, being 0 the value for black, and 255 the value for white, which in combination with different channels can generate multiple colors. Figure 2 illustrates a gray-scale image ($c = 1$) and its pixel representation. The zoom in Lenna’s eye show the pixels that compose that part of the image. The lower the number of pixels representing an object, the lower is the resolution of the image, and less natural the object represented looks.

Traditionally, image processing algorithms apply point operations, which replaces each pixel by a new value that depends on the original one. Depending on the technique used, different aspects of the image are highlighted, enabling its measurement. This processing also includes filtering, noise reduction, projections and several statistic analyses. However, such techniques pose limitations related to generalization. On the other hand, since the advent of AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012), the winner of the ImageNet Large-Scale Visual Recognition Challenge in 2012, Convolutional Neural Networks (CNNs) have shown to be

Figure 2 – Example of a gray-scale image representation in pixels. The zoom in Lenna’s eye show each component of the matrix of pixels that composes it. This image is represented in one channel only, hence only black and white levels can be shown. Picture adapted from [Gonzales and Woods \(2002\)](#).

robust models for feature extraction, image classification and segmentation, and object detection ([HE *et al.*, 2016](#); [SZEGEDY *et al.*, 2015](#); [DENG *et al.*, 2009](#)). These new state-of-the-art models in some sense also apply filtering, point-wise and statistical operations, but in a hierarchical and self-learning manner. As a consequence, they are more generalizable and are capable of distinguishing hundreds of different categories.

2.2 Deep learning

Deep learning is a specific type of machine learning where a computational model recognizes patterns and make inferences based on the hierarchy of concepts that split them in different levels of abstraction, what is characterized as representation learning ([GOODFELLOW *et al.*, 2016](#)). This hierarchical learning is defined by a composition of functions, where a function $f(x)$ is composed of several chained functions, one for each layer, enabling a model to learn successive intermediate representations of the input data. Figure 3 illustrates an example. The lower layers extract simple features such as texture, edges, and contours, while higher layers can determine complex concepts, such as a human face or a car wheel ([GOODFELLOW *et al.*, 2016](#)).

2.2.1 Deep Neural Networks

The functionality of the human brain inspires these computational models. They are made up of many artificial neurons, which are computational units that process an input and produce an output based on an activation function ϕ ([SEWAK; KARIM; PUJARI, 2018](#)). Common activation functions are the Sigmoid expressed in Equation 2.1, and the Rectified Linear Unit

Figure 3 – Representation of the feature extraction process in a deep neural network. The first layers retrieve properties such as corners and edges and last layers learn high level properties such as object parts, which are built from the hierarchy of features learned in the previous layers. Picture adapted from [Goodfellow *et al.* \(2016\)](#).

(ReLU) given by Equation 2.2. Note that both only produce positive values. These functions are designed like this to better highlight the patterns found while sweeping the image. Once a feature matches the kernel used, the pixels in that area are kept, while the others are discarded.

$$\sigma(z) = \frac{1}{1 + e^{-z}} \quad (2.1)$$

$$R(z) = \max(0, z) \quad (2.2)$$

When connected to other units, a neuron simulates neural synapses transmitting the information forward, which is called a feed-forward network. In this architecture, the data transmitted from a neuron to another is controlled by weights w ([SEWAK; KARIM; PUJARI, 2018](#)). These weights are parameters to the function $f(x)$ that calculates the desired output y . In a model defined by three layers, for instance, $f(x)$ is formed by three compositions of $f_r(x_r; w_r; b_r)$, where r indicates a layer, x is the matrix with the input image values, w is the function's weights, and θ is the bias array. In this context $f_1(x_1) = x_2$, $f_2(x_2) = x_3$ and $f_3(x_3) = y$ ([PONTI *et al.*, 2017](#)). The learning process is given by the estimation of the most suitable values of w for $f(x)$. The optimization is computed using the Gradient Descent method to find the local minimum of a differentiable function. This feeding forward process is expressed in Equation 2.3, where x_r is

the input for layer r , w_r the layer's weights, and θ is the bias array (MELLO; PONTI, 2018).

$$f(x) = \sum_{i=1}^r \phi_i(w_i x_i) + \theta \quad (2.3)$$

When these networks have more than 3 hidden layers, they are known as deep neural networks. The input layer receives x , the output layer produces y , and the intermediary layers, known as hidden layers, define the number of hyperplanes to shatter the input space (MELLO; PONTI, 2018). If an layer is fully connected, all g neurons in it are connected to all h neurons in the next layer, meaning g^h values have to be estimated, resulting in high computational cost. However, sparser models are available. They are known as Convolutional Neural Networks (CNNs).

2.2.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a particular case of deep neural networks that perform linear operations called convolutions in at least one layer. This operation for image inputs is defined in Equation 2.4, where K is a weighting function called kernel, I is the input image, i and j are rows and column indexes, and a and b are row and column shifts defined by the kernel size (GOODFELLOW *et al.*, 2016). These filters swipe the input image and transform it with a linear composition of the neighboring pixels. In this process, each layer learns a specific aspect of the image and transmits this data to the next layer in the chain. As a result, lower layers learn simple properties such as edges and contours, while the higher ones learn complex features formed by these properties composition, such as parts (PONTI; COSTA, 2018; LECUN; BENGIO; HINTON, 2015).

$$s(i, j) = (I * K)(i, j) = \sum \sum I(i - a, j - b) K(a, b) \quad (2.4)$$

CNNs also have multiple layers and are currently the most utilized approach to learn complex patterns in multidimensional data. This is because these networks filter the input images considering its multidimensionality by extracting the attributes of objects from local receptive fields through chained convolutions (GOODFELLOW *et al.*, 2016; PONTI; COSTA, 2018; PONTI *et al.*, 2017; LECUN; BENGIO; HINTON, 2015; FERREIRA *et al.*, 2018). In addition, convolution operations leverage three important properties for optimization: sparse interactions, parameter sharing, and equivariant representations. Additionally, it accepts inputs of variable size. These models are at the forefront of commercial applications of deep learning nowadays.

2.3 Unsupervised Learning

When all training variables are known, and predictions are based on labeled examples, the learning process is defined as supervised, or “learning with a teacher”. In these cases, the

computational model tries to define the probability density function $P(y|x)$ that better fits the training data, where y is the expected output and x the input. Therefore, since y is known, when an error has been made, the supervisor or “teacher” is able to indicate the correct value for the prediction. Differently, in unsupervised learning, y is not available, so the objective changes to directly inferring the properties of the probability density function $P(x)$ without a supervision that can indicate if a prediction is correct or not (HASTIE; TIBSHIRANI; FRIEDMAN, 2009). It can also be realized as a scalar-valued energy function $E(x)$ designed to produce low energy values (high probability) when x is similar to some training data examples, and high energy values (low probability) otherwise (RANZATO *et al.*, 2007).

To accomplish that, the learning process must be reconditioned. The goal shifts to capturing regularities from the data in order to extract useful representations or for restoring corrupted values (RANZATO *et al.*, 2007). Usually, it is given by the association or clustering rules that estimate $P(x)$ or $E(x)$ by evaluating how the space of features is organized. One common approach is to identify low-dimensional manifolds within the input space that represent the high data density, what may give information about the associations among the variables (HASTIE; TIBSHIRANI; FRIEDMAN, 2009). A manifold is a topological space where each point has its neighborhood locally homeomorphic to a subset of an Euclidean space, meaning that when projected, they preserve the topological properties of the original space, hence the inverse is continuous (RIAZA, 2008; MORITA, 2001). Examples of low-dimensional manifolds are lines and circles with one dimension, and surfaces such as planes, spheres and torus with two dimensions (MORITA, 2001). Examples of approaches that explore manifolds and its dimensionality reduction are principal components or curves, multidimensional scaling, and self-organizing maps.

On the other hand, the clustering approach, as well as mixturing modeling, try to find multiple convex regions of the input space that contains modes of $P(x)$, what may tell if the probability density function can be represented by a mixture of simpler densities for distinct types or classes (HASTIE; TIBSHIRANI; FRIEDMAN, 2009). The next sessions describe in more detail some of the unsupervised algorithms studied in this research.

2.3.1 Autoencoders

Autoencoders were first introduced by Rumelhart, Hinton and Williams (1986) to address the issue of “backpropagation without a teacher”. The authors propose to set the input as the supervisor by changing the training goal to transforming the input into output with minimum distortion (BALDI, 2012). In order to do that, the autoencoder is composed of three structures: a code z , defined as the latent space between input x and output $\hat{x}$, where the latent space is a space more representative than the feature space; an encoder A , responsible for encoding the input by predicting the best values for the latent code z ; and a decoder B , that reconstructs from the code

z the output $\hat{x}$.

$$\min_{A,B} E(A,B) = \min_{A,B} \sum_{t=1}^m E(x_t) = \min_{A,B} \sum_{t=1}^m \Delta(A(x_t) \circ B, x_t) \quad (2.5)$$

In this context, let's consider $\mathbb{F}$ and $\mathbb{G}$ sets, $\mathcal{A}$ a class of functions from $\mathbb{F}^v$ to $\mathbb{G}^s$, and $\mathcal{B}$ a class of functions from $\mathbb{G}^s$ to $\mathbb{F}^v$, where v is the input size, and s is the code size. The objective is to find $A \in \mathcal{A}$ and $B \in \mathcal{B}$ that minimize the energy function expressed in Equation 2.5, where Δ is a dissimilarity or distortion function defined over $\mathbb{F}^v$ (BALDI, 2012).

However, when minimizing the energy function described in Equation 2.5, one must assure that the model will not simply replicate the input. According to Goodfellow *et al.* (2016), when x and $\hat{x}$ are exactly the same, this model becomes less useful to machine learning applications, once the model will not be learning the innate properties of the feature space, hence the latent space will not be generated. The structure must be defined so that each training sample can be designed to be a unique code that should be reconstructed with minimum error. Moreover, unobserved data points must be associated to the most similar training sample (RANZATO *et al.*, 2007). In order to do that, some constraints are added. One possible approach is to add noise to the input and reconstruct the output without this noise, what is called denoising autoencoder. Another possibility is to add restrictions to the code to force the model into finding the main attributes that represent the input, what enables the autoencoder to learn specific properties of objects, what is called undercomplete autoencoder (KODIROV; XIANG; GONG, 2017). This is done by reducing the dimension of z , what consequently reduces the reconstruction error while attempting to learning an intrinsic dimension, or low-dimensional manifold, of the input data (RANZATO *et al.*, 2007; ALMOUSLI; VINCENT, 2013; CAVALLARI; RIBEIRO; PONTI, 2018).

An autoencoder can be coded in multiple forms. Kodirov, Xiang and Gong (2017), for instance, build this model based on the resolution of one Sylvester equation, calculated by the Bartels-Stewart algorithm (BARTELS; STEWART, 1972). Another common approach is to define the AE as a neural network. When it has a small hidden layer, it learns low-dimensional representations and behave similar to PCA. Although, when it has multiple hidden layers the encoder and decoder are no longer linear, what leads to manifold learning (RANZATO *et al.*, 2007).

These models have proven to be efficient for dimensionality reduction and feature learning. Recent studies use them as generative models since they can learn the distribution of data without supervision (MESCHEDER; NOWOZIN; GEIGER, 2017; GOODFELLOW *et al.*, 2016). One drawback of this approach is that when using Gradient Descent optimization, learning a deep autoencoder will take longer to converge to a appropriate low-dimensional representation (HINTON; SALAKHUTDINOV, 2006).

2.3.2 Principal Component Analysis (PCA)

This algorithm is often used for dimensionality reduction, for the study of data variance along reference axes, and for analyzing subspaces. As the autoencoder, this variance is analyzed in such a way that once the features are projected, the organization in space of a given point and its neighbors should be preserved from the source domain to the target domain. The PCA performs linear combinations from orthogonal vectors in the target space, simplifying linearly correlated dimensions and centralizing the representation (MELLO; PONTI, 2018). The goal of this mathematical model is to extract the most representative information of the input data and express it as a set of orthogonal variables called *principal components* (ABDI; WILLIAMS, 2010).

PCA is calculated based on the Singular Value Decomposition (SVD) of rectangular matrices. First, it centralizes and standardizes the data, then it computes the covariance matrix and its *eigenvectors* and *eigenvalues*. The *eigenvectors* are called principal components, where the first one is always the component with highest variance, hence the most important. The following components obey this ranking logic, where the last one has the lowest variance. The *eigenvalues* are the sum of the square of the distances that determine the best fitting line that combines two dimensions (eigenvector), indicating the strength of the association of each attribute to its corresponding component (HASTIE; TIBSHIRANI; FRIEDMAN, 2009; MELLO; PONTI, 2018). It is most useful when lies on or is close to a linear subspace of the dataset, since PCA finds a basis for this linear subspace disregarding the irrelevant features (CAYTON, 2005).

2.3.3 Manifold Learning

These algorithms can be viewed as non-linear analogs to PCA. Instead of decomposing the data into linear subspaces, they decompose it into manifolds, so they are often used for nonlinear dimensionality reduction. The main idea behind this approach is that the dimensionality of the data is artificially high, and that data points are samples from a low-dimensional manifold embedded in a high dimensional space (CAYTON, 2005). Although having different approaches, in some scenarios those were found to be similar to PCA in terms of empirical results (NSIMBA; LEVADA, 2019).

Many algorithms were developed to address the manifold problem. In the next sessions we discuss two approaches used in our experiments: *Isomap* and *Locally Linear Embedding*. Note that both require a neighborhood-size parameter k , and that the algorithm assumes that the manifold is approximately flat within each neighborhood.

2.3.3.1 Isomap

Isomap is short for isometric feature mapping and consists of two main steps: 1) estimate the distances along the manifold using shortest-path distances on the k -nearest neighbor graph

of the dataset; and 2) find the points in the low-dimension Euclidean space whose interpoint distances match the ones found in step 1 (CAYTON, 2005).

The distances estimated in step 1 are called geodesic distances. To find them, the k -nearest neighbors are represented in a weighted graph where the shortest distance between each pair of data sample is defined using a Dijkstra's or Floyd's algorithm. Then, Multidimensional Scaling (MDS) (COX; COX, 2008) is applied on the set of the square distances to find them in a Euclidean space. Consequently, it automatically provides an estimate of the dimensionality of the underlying manifold given by the number of non-zero eigenvalues found by MDS (BERNSTEIN *et al.*, 2000; CAYTON, 2005).

One drawback of Isomap is that it can only recover the parametrization when the manifold is isometrically embedded and compact, and the parameter space is convex and well sampled everywhere (no holes). Although, since MDS is continuous, small changes in the input will result in small changes in the output, what makes the algorithm robust to perturbations (CAYTON, 2005).

2.3.3.2 Locally Linear Embedding (LLE)

LLE can be seen as a series of local PCAs globally compared to find the best non-linear embedding (ROWEIS; SAUL, 2000). This approach computes low dimensional neighborhoods preserving embeddings of high dimensional data. It exploits local symmetries of linear reconstructions to discover the nonlinear structure in high dimensional data by mapping the inputs into a single lower dimension global coordinate system (SAUL; ROWEIS, 2000).

Saul and Roweis (2000) defines the algorithm in three steps: 1) compute the k neighbors of each data point; 2) calculate the weights that best reconstruct each data point; and 3) compute the vectors best reconstructed by the weights. Each high dimensional sample is mapped to a low dimensional vector that represents global external coordinates of the manifold.

Locally Linear Embedding expects that each data point and neighbors lie on or are close to a locally linear patch of the manifold. The geometry formed is invariant to rotations, rescaling and translations and is characterized by linear coefficients that reconstruct each data point from its neighbors (SAUL; ROWEIS, 2000; ROWEIS; SAUL, 2000).

2.4 Zero-Shot Learning

The zero-shot learning problem comes from the intuition that humans are able to identify and recognize objects without visual samples (KODIROV; XIANG; GONG, 2017). They can discriminate around 30 thousand different objects and understand underlying concepts from visual representations based on prior knowledge, which allows them to recognize unseen objects when provided high-level descriptions (LAMPERT; NICKISCH; HARMELING, 2009; XU *et*

al., 2018). For example, a child can identify a “leopard” if he/she has seen cats before and know that leopards are like big yellow cats with black spots. In this case, the image of a cat represents the knowledge obtained from the visual domain, and the concept of colors and texture comes from the semantic characterization abstracted from classes that share the same features, such as dalmatians.

The idea is motivated by the fact that collecting image samples for rare and fine-grained databases is difficult, and from the limitation of image capturing, annotation and categorisation of thousands different objects (FARHADI *et al.*, 2009). Therefore, zero-shot learning relies on the existence of a labeled training set of seen classes X , and the knowledge about how the unseen classes X' are related to the seen classes, which could be acquired from experience just like humans (KODIROV; XIANG; GONG, 2017).

According to Farhadi *et al.* (2009), meaningful high-level concepts transcend class boundaries, so even though X and X' are disjoint, they can share semantic aspects. In this context, let us consider the sets of visual attributes $V \in X$ and $V' \in X'$ extracted from images as described in Section 2.1.2, and the sets of semantic prototypes $S \in X$ and $S' \in X'$ embedded in vectors as described in Section 2.1.1. A zero-shot learning method will learn a function f_{zs} that can be defined in three forms:

- $f_{zs} : V \rightarrow S$: this method learns a projection function from the visual to the semantic space. At training time the features from V are regressed or embedded in the S domain. At test time V' is used to estimate S' .
- $f_{zs} : S \rightarrow V$: this method is the opposite of the previous one. It learns from the semantic feature space to the visual one. At training time the features from S are projected to V feature space. At test time S' is used to estimate V' .
- $f_{zs} : V \oplus S \rightarrow K$: this last approach learns a combination of both visual and semantic domains creating a new feature space K with the merged projection. Differently from the previous methods, at test time these algorithms have to estimate the new prototypes to compare each example to, since the latent space is neither equal to V' or S' .

The classification of the unseen classes is based on a nearest neighbor search that looks for the examples in the known domain closest to the examples estimated by the function f_{zs} . For example, when using the first approach, the class of an unseen example $x \in V'$ will be defined based on the prototype $p \in S'$ that is closest to $f_{zs}(x)$. However, this process can suffer from the project domain shift problem, which comes from the fact that seen and unseen classes may have a different visual appearance of the same attributes. Since the projection is learned only from the training classes, the latent space formed for the test classes (the unseen ones) might be misplaced (KODIROV; XIANG; GONG, 2017; FU *et al.*, 2015). To mitigate that, the algorithm can learn a merged embedding using $f_{zs} : V \oplus S \rightarrow K$ approach, or adding more constraints to the training,

as the autoencoder presented by [Kodirov, Xiang and Gong \(2017\)](#) that learns both $f_{zS} : V \rightarrow S$ and $f_{zV} : S \rightarrow V$ together. More details of how this evaluation is performed are given in [Chapter 3](#) and in [Section 4.5.3](#).

RELATED WORK

Even before the advent of CNNs, there were studies on how to boost image classification with the addition of semantic features in the learning model. In [Farhadi *et al.* \(2009\)](#), the authors used attributes for object description, applying L1-regularized logistic regression to better select semantic features in bounding-box restricted objects, and by doing so, achieved cross-category generalization, avoiding datasets training biases. Semantic attributes were also used to classify unseen classes and apply probability classifiers to show that shared semantic concepts can be learned from different categories, motivating zero/one/few-shot learning ([LAMPERT; NICKISCH; HARMELING, 2009](#)). In terms of both visual and semantic features, [Deselaers and Ferrari \(2011\)](#) showed that a function that exploits visual and semantic attributes surpasses models based solely on image features. Moreover, it demonstrated that semantic similarity alludes to visual similarity, indicating that computer vision algorithms can classify semantically meaningful classes. Also, [Su and Jurie \(2012\)](#) use SVMs combined to BoW histograms to build visual-semantic image descriptors by exploiting local and global regions.

Since CNNs became the state-of-the art in image and vision tasks, studies employed semantic attributes with deep networks to address more challenging scenarios such as clutter or pose variation and tasks such as zero-shot learning and multilabel classification. In this sense, [Ren *et al.* \(2015\)](#) explore semantic attributes by mapping semantically meaningful subregions of an image to their labels for multilabel image annotation and zero-shot learning applications. Similarly, [Wang *et al.* \(2016\)](#), [Zhang *et al.* \(2018\)](#) use a CNN to acquire visual features and a RNN to capture the latent semantic dependencies of each region in an image, which increases performance on predicting small objects and visual concepts. On the other hand, [Juan *et al.* \(2019\)](#) associate graph relations with visual attributes to capture semantics close to human perception by learning features capable of associating images to query strings for image retrieval. Meanwhile, [Xu *et al.* \(2018\)](#), [Lin, RoyChowdhury and Maji \(2015\)](#) use the association of two CNNs to leverage semantic characteristics and combine them with visual attributes, projecting the feature space in an end-to-end framework for classifying CUB200-2011 dataset.

In the context of zero-shot learning, [Lu \(2015\)](#) uses both the visual and semantic domain to project data to the latent space using PCA/ICA algorithm in the Softmax output of GoogleNet ([SZEGEDY *et al.*, 2015](#)). With that, that study aims to extract visual attributes shared by similar classes and use Canonical Correlation Analysis (CCA) ([HOTELLING, 1992](#); [HARDOON; SZEDMAK; SHAW-TAYLOR, 2004](#)) to combine semantic and visual representations for ZSL. The author indicates that the PCA/ICA components can be seen as visual features shared by similar object classes. The idea consists in exploiting the fact that the Softmax layer of a CNN holds much more information than the class itself. For example, the probability of a tiger being classified as a lion is much higher than it being classified as a chair. [Lu \(2015\)](#) defines the visual similarity as the cosine of the PCA or ICA components, while the semantic similarity is computed over the shortest path between two classes on the WordNet graph. At training time, he uses CCA to learn a bridge between these two domains projecting these attributes in a new feature space. At test time, the author measures the cosine similarity of the top k predictors to the prototypes to estimate the class of an image. Using this approach, [Lu \(2015\)](#) reached state-of-the-art results for ZSL classification on ImageNet and concluded that one should build a bridge between visual and semantic representation to achieve consistent mapping of these two domains.

The works of [Kodirov, Xiang and Gong \(2017\)](#) and [Zhang and Saligrama \(2016\)](#) were used as comparison baselines. The former one inspired the pipeline of the method developed and is the basis of classification and zero-shot learning for performance comparison. The latter is a baseline for zero-shot learning only once it uses a similar approach to the one developed here. The next sections discuss in more detail each one of these works.

3.1 ZSL using an autoencoder

Differently from traditional zero-shot learning methods, [Kodirov, Xiang and Gong \(2017\)](#) use an autoencoder to associate visual and semantic features. The algorithm, called SAE for short, learns intrinsic relationship of the semantic data, among seen and unseen classes, using visual data extracted from InceptionV1 ([SZEGEDY *et al.*, 2015](#)) as input, and semantic attributes in the loss definition of the latent space, which transforms the AE in a supervised learning model. The main idea behind SAE is that the decoder part of the auto-encoder adds another constraint to the training of the model, forcing it to reconstruct the visual features from the code embedding, hence mitigating the project domain shift problem described in Section 2.4. This problem states that the visual properties of the same attributes can be different in seen and unseen classes, what can misplace the projection of these features at test time. By forcing the model to learn the projection of one domain to another, and then learn how to project the features back to the original ones, one is reducing the learning bias and increasing generalization, overcoming this way the project domain shift issue.

This approach, shown in Figure 4c, contrasts typical ZSL methods that usually learn the projection function from the visual space to the semantic space, as displayed in Figure 4a, or vice-versa, as illustrated in Figure 4b.

Figure 4 – Representation of the model developed by [Kodirov, Xiang and Gong \(2017\)](#). The authors use an autoencoder that encodes the visual features (X) in the semantic space (S), and decodes it back to the visual space. With this approach they were able to mitigate the project domain shift problem. Picture adapted from [Kodirov, Xiang and Gong \(2017\)](#).

[Kodirov, Xiang and Gong \(2017\)](#) developed their method from a simple and symmetric auto-encoder. The model has as input visual features, the latent space defined by the semantic attributes, and the output defined by the same visual features as the input. The authors optimized this model training to computing a Sylvester equation, calculated by the Bartels-Stewart algorithm ([BARTELS; STEWART, 1972](#)), which solves an equation of type $AW + WB = C$. For [Kodirov, Xiang and Gong \(2017\)](#), W defines the matrix of weights, A the semantic space (SS^T), B the visual space (λXX^T), and C the visual-semantic projection ($(1 + \lambda SX^T)$). With that, their algorithm outperforms all compared methods for zero-shot learning showing high computational training time performance, due to the simplicity of the model and its fast convergency. In addition, their approach demonstrates viability for clustering applications.

3.2 ZSL by merging visual and semantic data

On another aspect, [Zhang and Saligrama \(2016\)](#) built a class-independent classifier that jointly learns the semantic and visual parameters of these domains to predict if a pair of source and target instances belong to the same class, intrinsically performing ZSL. For that, the authors use a joint discriminative learning framework, here called JDL, based on a dictionary that learns parameters of both domains together by mapping data instances into a newly defined latent space, and formulate a empirical risk function to measure the similarity of this mapping that minimizes the regularized hinge loss of the joint latent embedding. The goal here is to define if two examples taken from different domains belong to the same class or not, an by doing that defining zero-shot learning as a binary problem. This idea is represented in Figure 5.

At training time, the algorithm learns a low-dimensional latent embedding to represent the visual and semantic domains that is capable of inferring cross-domain statistical relationships. It

Figure 5 – Representation of the model developed by [Zhang and Saligrama \(2016\)](#). The authors define the zero-shot learning problem as a binary problem, where the model aims to identify if the source and target domains examples belong to the same class or not. They do that by mapping these classes in a dictionary. Picture adapted from [Zhang and Saligrama \(2016\)](#).

also learns a scoring function that takes an arbitrary instance from the source domain (semantic), and another from the target domain (visual), and compute the likelihood of the pair belonging to the same class, defining this way ZSL as a binary classification problem. It is important to highlight that at training time it knows the source domain vectors of unseen classes to build the prototypes. At test time, [Zhang and Saligrama \(2016\)](#) compute the joint embeddings and use the similarity function for estimating the probability of the likelihood and image belonging to the same class.

METHOD

4.1 Model Formulation

We propose and evaluate the application of an undercomplete autoencoder to combine semantic and visual data into one lower dimension representation. The idea is that this structure will learn the underlying arrangement of the domain of both attributes, improving class separability in the latent space, with the need for fewer examples in training data, a result expected based on the curse-of-dimensionality (FRIEDMAN, 1997) and dimensionality reduction (CAYTON, 2005) hypotheses. In this sense, let us consider an image $I \in \mathbb{N}^{m \times n \times 3}$ and two functions capable of obtaining features from I : (i) a function $f_v : I \rightarrow \mathbb{R}^\alpha$ that retrieves visual features $V \in \mathbb{R}^\alpha$, and (ii) a function $f_s : I \rightarrow \mathbb{R}^\beta$ that collects the semantic attributes $S \in \mathbb{R}^\beta$. We posit that exists a fusion function, given by Equation 4.1 that combines V and S in a way where the new lower dimension space Z is more distinctive for classification tasks.

$$f_f : V \oplus S \rightarrow \mathbb{R}^\delta = Z \quad (4.1)$$

The encoder part of our autotoencoder defines f_f expressed in Equation 4.1, where $\oplus$ represents the concatenation of the input domains, Z is the latent space the encoder forms, and V and S are inputs to it, where V consists of visual features extracted from CNNs, and S is a set of semantic attributes either described by humans or detected from computational models. S can characterize a specific category in the dataset or a single image. To accommodate scenarios where features are automatically retrieved, f_f must be robust to noise once these extractors are subject to errors.

Our model is called *Visual Semantic Encoder*, or VSE for short. It is trained to identify the most relevant features of V and S building a structure Z smaller than these inputs. According to Ranzato *et al.* (2007), Almousli and Vincent (2013), by reducing the dimension of Z , we can reduce the reconstruction error while attempting to learning an intrinsic dimension of the input

data. Therefore, $\delta < \alpha + \beta$, where δ is the number of components in the latent space, α in the visual space, and β in the semantic space. In this project, we set $\delta = \beta$, which allows a direct comparison to Kodirov, Xiang and Gong (2017).

Figure 6 – The structure of VSE for an autoencoder with three layers in the encoder and three layers in the decoder. The number of neurons follow Equation 4.2. κ values are highlighted in red.

In order to guarantee not only a latent space smaller than the input but also a smaller dimension of the intermediary layers of the neural network, we dynamically set the number of neurons in each layer based on δ , what forces the encoder to a funnel shape, meaning each layer has a smaller size than the previous layer. The same applies to the decoder but in a mirrored manner, making it the inverse of a funnel. The number of neurons in each layer is defined by Equation 4.2.

$$\rho = s_i - \kappa(s_i - 2\delta), \quad (4.2)$$

where s_i is the input size ($\alpha + \beta$), δ is the encoding size, and κ is a reduction factor defined based on the depth of the layer. $\kappa = d/L$, where L is the number of layers in the encoder or decoder structures, and d is the depth of the layer. For an encoder of three layers, κ is equal to 0.3 in the first layer, 0.6 in the second, and 0.9 in the third. While for the decoder, κ is equal to 0.9 in the first layer, 0.6 in the second, and 0.3 in the third one, as illustrated in Figure 6.

4.2 Feature extraction

Feature extraction is performed in two different steps, one for each input domain. Visual features are retrieved using Convolutional Neural Networks, and semantic features can be acquired from a set of descriptions given by specialists, as well as automatically from off-the-shelf models such as RCNNs (ZHANG *et al.*, 2018), object detection networks (REDMON; FARHADI, 2018), or even text mining algorithms such as (MIKOLOV *et al.*, 2013a; MIKOLOV *et al.*, 2013b). These sets of data are then normalized, concatenated, and introduced to VSE. Extraction steps are illustrated in Figure 7.

Figure 7 – Illustration of feature extraction, training, and classification processes. The upper left shows how visual features are acquired, and the lower left how the semantic descriptors are built. Both these data are normalized, concatenated, and then input to VSE. After training, the code generated is classified by a linear SVM. Note that classes are not used in the training process.

4.2.1 Visual Features

The visual attributes are obtained from CNNs, which extract the most representative features of each image. We analyzed and tested two different architectures: (i) Inception, for which we evaluate results on Inception V1 (I1) (SZEGEDY *et al.*, 2015) and Inception V3 (I3) (SZEGEDY *et al.*, 2016); and (ii) ResNet, where we study ResNet50 (RN) (HE *et al.*, 2016). For all cases, we use the attributes selected by the last average pooling layer of the neural network trained on *ImageNet* dataset, as shown in Figure 7, where *FC1000* is the fully connected layer responsible for estimating the classes. According to Ge *et al.* (2017), Xian *et al.* (2018b) and Yosinski *et al.* (2014), this layer is widely used because it provides high-level global features, combining low-level local features such as shape, edge, and color throughout the hierarchical structure of the model. This step is represented in the upper left part of Figure 7. An RGB image is input to the network that extracts key features in each layer and combines them all in the layer immediately before *softmax*. These features are normalized using L2 normalization, which rescale the visual attributes to have Euclidian norm equal to one, and then are input to VSE.

Inception V1, also known as GoogLeNet, is a 22 deep Convolutional Neural Network introduced by Szegedy *et al.* (2015) in the ImageNet Large-Scale Visual Recognition Challenge 2014 (ILSVRC14). This architecture was developed contrasting the idea of merely increasing the size of networks and its width to improve its recognition performance. Aiming to limit the number of parameters and consequently the computational costs, the authors propose a design with more sparse layers, and introduce the concept of Inception modules with dimensionality reduction. These structures stack convolution scales to take advantage of different cluster sizes, which improves the model’s discriminative power. The receptive field of this network has size 224x224 in the RGB color space with mean equal to zero. It is built from chained Inception modules where convolutions use rectified linear activation (ReLU) (NAIR; HINTON, 2010). Its

fully connected layer has 1024 neurons and is used to extract the visual features we input to VSE.

Because only two of our datasets have its visual features available in this setting, we had to extract the features of the others using the closest available model, which is Inception V3 (SZEGEDY *et al.*, 2016). This model is the enhancement of previous versions, a bit larger and with more stacked convolutions in the Inception modules, but all carefully arranged to avoid a high increase in computational costs. Differently from V1, Inception V3 spams 2048 features at its last average pooling layer.

On the other hand, ResNet models range from 18 to 152 layers, being the latter the winner of the ILSVRC challenge in 2015. According to He and Sun (2015), Srivastava, Greff and Schmidhuber (2015), raising the depth of a suitably deep model increases its training error. However, He *et al.* (2016) accommodate a large number of layers for ResNet by introducing a *deep residual network* framework, consequently avoiding accuracy saturation and degradation. This strategy consists in defining building blocks as $y = F(x, W_i) + x$, where x and y are the input and output vectors, and $F(x, W_i)$ is the residual mapping. In practice, this means that the previous layers' input is directly used to define the mapping of the current layer. This formulation was driven by the idea that if an identity mapping is optimal, the weights of the multiple linear layers approximate to zero, what is rare to occur, but the residual mapping proves to help in preconditioning the problem. In our experiments we use ResNet with 50 layers. The network's receptive field has size 224x224 in the RGB color space, and its fully connected layer has 2048 neurons. Convolutions' activation functions are ReLu (NAIR; HINTON, 2010).

4.2.2 Semantic Features

The semantic characteristics are defined based on each subject's attributes and can be visually related, such as color, texture, shape, and other physiognomy features; or behavioral related, such as environment, feeding, movement, and activities (FARHADI *et al.*, 2009). Figure 7 lower left illustrates the acquisition process of these data. They are also normalized using L2 norm. These features can be represented by binary or continuous values, but continuous data better describes the semantic domain (AKATA *et al.*, 2015; XIAN *et al.*, 2016), so we opt to use it when available, otherwise we use binary descriptors.

For Animals with Attributes (AwA) (XIAN *et al.*, 2018a) and CUB-200-2011 (WAH *et al.*, 2011), we employ features from specialists' annotations, which poses a problem because these attributes do not reflect real-world applications. These annotations are directly linked to labels and characterize an object within an image based on its general features. A zebra, for instance, is white with black stripes and has four legs. Since all zebras have these features, all images with this label will have these attributes, even if these properties are not present.

On the other hand, aPascal&Yahoo (aP&Y) (FARHADI *et al.*, 2009) and SUN Attributes

(SUN) (PATTERSON; HAYS, 2012) datasets have their semantic features retrieved using Amazon’s Mechanical Turk crowdsourcing marketplace (VOGEL; SCHIELE, 2004), hence each image has a unique semantic representation. However, annotating thousands of images is costly. To mitigate that, real-world applications often use automatic detectors, but they can be subject to errors originated from noisy data or misclassification, producing imprecise feature vectors.

To investigate the robustness of our approach, we simulate the degradation of the semantic data by randomly replacing the values of each feature vector. This process follows the steps described in Algorithm 1. We aim to substitute these features in the most arbitrary manner possible, so both the values and indexes to replace are randomly chosen. For instance, considering a degradation rate $r = 0.2$ and a semantic descriptor of 85 attributes, 16 random values of the semantic array are superseded by random values picked within the original distribution. With this, each image in S will end up having a unique set of semantic attributes, losing direct relation to its label but still keeping some descriptions of high-level concepts that can help VSE to build a better representation of the input image.

Algorithm 1 – Semantic Degradation Algorithm

Input: Semantic descriptors; Degradation rate

Output: Degraded semantic descriptors

$S \leftarrow$ semantic descriptors of shape $M \times N$

$r \leftarrow$ degradation rate $\in \mathbb{R}_{0..1}$

$a \leftarrow \max(S)$

$b \leftarrow \min(S)$

for each image descriptor $i \in S$

$\mathbf{r} \leftarrow N \times r$ random indexes $\in \mathbb{N}_{0..N}$

for each index $j \in \mathbf{r}$

$S[i, j] =$ random value from uniform distribution with $\max = a$ and $\min = b$

4.3 Model Training

Similarly to Kodirov, Xiang and Gong (2017), we define the *Visual Semantic Encoder* in a symmetric structure, which means that the same number of layers used in the encoder will be used in the decoder. The model receives a combination of the visual and semantic descriptors as input, which are normalized and then concatenated. The output is defined to be the same as the input. Once VSE is trained, the generated embedding is used to classify the code built. The end-to-end process can be observed in Figure 7.

Regarding the loss of the model, Kodirov, Xiang and Gong (2017) defines the latent space as the same as the semantic space, and the loss of the autoencoder as the difference between the code formed and the expected semantic features. Differently from them, the loss of VSE is defined as the reconstruction error of the model. This error is computed by the *Mean Square Error* between input and output. As stated in equation 4.1, input is defined as the concatenation

of visual and semantic arrays into one larger array. This concatenation happens before the model is trained and after both inputs have been normalized. Consequently, VSE has only one input array with $\alpha + \beta$ attributes, and the same is expected in the output. Loss is described in Equation 4.3, where $Y_i = V \oplus S$, and Y_o is the estimated output.

$$loss = \frac{1}{(\alpha + \beta)} \sum_{i=1}^{(\alpha+\beta)} (Y_i - Y_o)^2 \quad (4.3)$$

In this work, we consider unsupervised learning a model that is trained independently from its labels. Therefore the training loss is not linked to the final expected result, it is just a mean to get to the classification. In this sense, since the semantic data is part of our loss and somehow linked to each image's category, VSE is not entirely disconnected from labels. On the other hand, the code formed Z is not tied to the semantic input as in Kodirov, Xiang and Gong (2017), so our approach cannot be characterized as a supervised algorithm. Moreover, assuming that the semantic information can be retrieved from off-the-shelf models, VSE falls under the concept of unsupervised learning since these models only indicate what could be within an image, which makes the algorithm lose direct relation to a specific label.

The VSE was coded in Keras (CHOLLET, 2015) and trained for 50 epochs in batches of 256 examples. The dataset is split into training and test sets in an 80 by 20 proportion. SVM classification is employed at the end of each epoch, without kernel (a.k.a. linear) and with miss-classification margin (or cost) $C = 1$. The model's loss is defined as the *Mean Square Error* of the reconstruction with *Adam* optimizer, and each layer's activation function is a *ReLu* (NAIR; HINTON, 2010). The source code can be found in <https://github.com/MAP-VICG/VisualSemanticEncoder>.

4.4 Datasets

In our experiments, we used four benchmark datasets for evaluation: Animals with Attributes (AwA) (XIAN *et al.*, 2018a), CUB-200-2011 (WAH *et al.*, 2011), aPascal&Yahoo (aP&Y) (FARHADI *et al.*, 2009), and SUN Attributes (SUN) (PATTERSON *et al.*, 2014); and one for transfer learning: ImageNet (DENG *et al.*, 2009), which is solely used for training the CNNs that extract the visual features.

- **Animals with Attributes:** there are two versions of AwA, both with 50 classes of animals and 85 semantic attributes: AwA1 with 30,475 images, and AwA2 with 37,322 images. Although AwA1 was discontinued, we used it in order to compare our results with Kodirov, Xiang and Gong (2017). This dataset is relatively easy to classify since classes are very distinct. Moreover, different animals share common attributes, so this data is commonly used to evaluate *zero-shot-learning* classification applications.

- **CUB-200-2011**: this dataset has 11,788 images, 200 classes, and 312 semantic attributes. It is a fine-grained dataset, so there is significant class overlap and intra-class variability, making classification more challenging (XU *et al.*, 2018). This dataset is also widely used for *zero-shot-learning*.
- **SUN Attributes**: it is a large-scale scene attribute database built by Patterson and Hays (2012) on top of the diverse SUN categorical dataset (XIAO *et al.*, 2010). It spans 717 different categories on 14,340 images. The semantic attributes were also retrieved by Amazon’s Mechanical Turk (VOGEL; SCHIELE, 2004) resulting in a taxonomy of 102 discriminative attributes. This benchmark is used for high-level scene understanding and fine-grained scene recognition. Scenes contain a high variety of objects and semantic attributes include the description of materials, functions, spatial features and object presence.
- **aPascal&Yahoo**: this dataset was developed by Farhadi *et al.* (2009), where 64 semantic attributes were detected using Amazon’s Mechanical Turk crowdsourcing marketplace (VOGEL; SCHIELE, 2004). The authors analyzed 20 object classes of PASCAL VOC 2008 dataset (EVERINGHAM *et al.*, 2008), and 12 objects retrieved from the images they have downloaded from Yahoo! search engine. VOC2008 has 12,695 examples, while Yahoo has 2,644. Classes include animals, vehicles, and indoor objects such as mugs and chairs. Unfortunately, VOC2008’s webpage is no longer available, and standard repositories only host VOC2007 and VOC2012 data. Therefore, we used a subset of VOC2012 data with VOC2008’s images, a total of 6,340 examples.
- **ImageNet**: it is a dataset organized in hierarchies where each node has an average of 500 images. It has more than 14 million images labeled with words or sentences. Due to its complexity and wide variability, this dataset is commonly used in transfer learning applications. It is also the base of the ImageNet Large-Scale Visual Recognition Challenge.

Table 1 indicate the number of images, categories and semantic attributes for each database used for classification. Features extracted can be found in <https://github.com/MAP-VICG/VisualSemanticEncoder>.

Base de Dados	# Imagens	# Categorias	# Atributos Sem.	Tipo de Atributo Sem.
AwA1	30,475	50	85	contínuo
AwA2	37,322	50	85	contínuo
CUB	11,788	200	312	contínuo
SUN	14,340	717	102	contínuo
aP&Y	8,984*	32	64	binário

Table 1 – Resumo do número de imagens, categorias e arranjo dos atributos semânticos para as bases de dados AwA, CUB, SUN e aP&Y. Os valores com (*) não são os mesmos observados na literatura.

Figure 8 show some examples for AWA and CUB datasets. They have the same attributes' set for all images belonging to the same class, regardless of the perspective and background. One can observe that images of the same classes are taken from different perspectives; consequently, important features are not always present. Moreover, confusing objects can appear, making classification more difficult. Note the human with the german shepherd in the first picture, and a finger pointing to a hamster. Text may also be present, such as in the first picture of the sooty albatross, the bird in blue background.

Figure 8 – Images from AWA2 on the first 4 columns (left to right) from classes: german shepherd, zebra, humpback whale and hamster, and from CUB on the last 4 columns from classes: sooty albatross, glaucous-winged gull, chestnut-sided warbler, and brewer blackbird. Note images from the same class may be captured from different perspectives, may have confusing objects, clutter and varying resolution.

On the other hand, images from SUN and aP&Y databases have a great variability of features in both visual and semantic domains. As said before, for these datasets, we have a unique semantic array for each image with a description of materials, functions, spatial features, or object presence. Therefore variations in perspectives are taken into account. Figure 9 show some examples. Note that images from the same scene may contain different elements and that the same image can have more than one valid object for classification.

Figure 9 – Images from SUN on the first 4 columns (left to right) from classes: dock, embassy, kitchen and park, and from aP&Y on the last 4 columns from classes: aeroplane, person, mug, and centaur. Note that some scenes may not share the same set of attributes, that images can contain other valid classes within it, such as the person holding a cat and a mug with the picture of a cow, or even a mixture of different classes such as a centaur (person/horse).

4.5 Evaluation

We evaluate the performance of VAE in three different aspects: (i) space separability, that resumes to empirically analyzing the distribution of examples in the latent space; (ii) SVM classification, that provides a quantitative measure of how separable the latent space is; and (iii) Zero-Shot-Learning classification, that measures the viability VAE in an application that commonly uses semantic data. Next sections details each approach.

4.5.1 *Space Separability*

This evaluation uses the PCA algorithm to investigate how the arrays of features are spread in space regarding the visual, semantic, and merged domains. We understand that visual and semantic features hold different distributions and complexity in terms of the number of attributes, energy, and entropy, reflecting on the density of the clusters formed. To visualize that, we plot the two main components of the decomposition of PCA. With that, we expect to see clusters of similar classes close to one another, a more significant entropy in the visual space compared to the semantic one, and merged aspects of both domains in the space of characteristics formed by Z. We also evaluate how different semantic features influence the formation of these clusters and expect distinct arrangements for each set of features chosen.

In order to quantitatively observe how well-formed the clusters of classes are, we validate their consistency by measuring the Silhouette Score (ROUSSEEUW, 1987) of the feature space. This metric gives an idea of the cohesion of a cluster by identifying how similar a sample is to its own cluster, as well as an idea of separation by measuring how distant this sample is from the other clusters. This value ranges from -1 to $+1$. The higher the value the better is the match of a sample to its own cluster, and the worst is the match to other clusters. In the results, we observe the mean of the Silhouette Score over all samples.

4.5.2 *Linear Classification*

With this evaluation, we aim to observe quantitatively how linearly separable the merged space is. It consists of an ablation study of the method developed compared to SAE, the state-of-the-art algorithm proposed by Kodirov, Xiang and Gong (2017). For that, we use the SVM probabilistic classifier, which represents each sample as a point in space and aims to map them defining regions limited by linear borders (CORTES; VAPNIK, 1995). Since it has low bias and a lower bound on learning convergence, the accuracy on the test set can be used as a proxy measure for the class separability of the learned space (CORTES; VAPNIK, 1995; MELLO; PONTI, 2018).

The SVM model comprises a linear kernel with a miss-classification margin $C = 1.0$. It is essential to highlight that the classification step is not part of the network’s loss, it is only used

to evaluate the latent space formed at each epoch, so the model’s training is independent of the dataset’s labels, as shown in Figure 7.

The classification accuracy is determined by the mean of 5-fold stratified cross-validation classification, computed with the Sklearn metric *balanced accuracy score*, that gives the accuracy recall for each class (BRODERSEN *et al.*, 2010). For that, the analyzed database’s examples are shuffled and the split in a 80 by 20 proportion 5 times, keeping the classes’ distribution as close to the original one as possible.

4.5.3 Zero-Shot Learning

This last metric is used to evaluate performance of VAE for the classification accuracy of an application that commonly uses semantic data to overcome challenges imposed by the lack of examples at training time. In this case, our baseline is also SAE (KODIROV; XIANG; GONG, 2017), which was developed for ZSL mainly. Since this application expects disjoint training and test sets, here we use the splits based on the recommendation proposed by Brodersen *et al.* (2010), but also apply each algorithm 5 times. Because SAE is deterministic, we will observe no variation in the results. However, VSE’s performance may change based on how the weights are initialized, so we can see a minor oscillation for each try.

Differently from SVM, an unseen test label is given by a nearest neighbour search. For that, we estimate prototypes that represent each image or class and assign to the image being analyzed the label of the example closest to it. Since Kodirov, Xiang and Gong (2017) projects the visual space into the semantic space, the prototypes are the semantic data itself. Therefore, in the case of AWA and CUB datasets, all examples of one class will have the same semantic definition, while for SUN and aP&Y each image has its own unique representation. In our case, since we project the semantic and visual spaces into a new domain, we have to build our own prototypes; therefore, all databases studied have a unique representation for each image. Accuracy is then determined based on the number of examples correctly classified, and differently from *balanced accuracy score*, it does not consider the class distribution of the dataset. This may pose a problem for extremely unbalanced databases, but we opt to use it since it is the same as the one implemented by Kodirov, Xiang and Gong (2017).

RESULTS AND DISCUSSION

5.1 Results

The experiments were performed in the 5 datasets summarized in Table 1. VSE's training follows the steps mentioned in Section 4.3, and evaluation metrics used are the ones described in Section 4.5. Details of the results are given in the next sections.

5.1.1 Space separability

Figure 10 displays the difference between visual, semantic, and merged representations. PCA was applied to visual features extracted by ResNet50 on Animals with Attributes 2 dataset, as well as the 85 semantic features indicated by specialists, and the 85 attributes formed by VSE. In each domain, the attributes of each sample array was reduced to its two main components. The figure also shows the mean of the Silhouette Score for each feature space analyzed.

Figure 10 – PCA 2D plot of visual, semantic and VSE features. This image illustrates how separable a subset of classes of AwA2 is in each domain. The robustness of the space formed can also be observed with the Silhouette Score. We show terrestrial and aquatic animals, where terrestrial animals can be felines, equines or a dalmatian. The later was introduced as a confusion category. One can observe that the merged domain formed by VSE preserves the semantic and visual relations in space, while improving the cohesion and separability of the clusters.

To illustrate the difference between these spaces of characteristics, we chose twelve classes that belong to two distinct groups: terrestrial animals and aquatic ones. Also, terrestrial labels can be separated into two sub-groups: felines and equines. It is expected that the clusters of animals of each group remain close to one another.

We can observe in Figure 10 a higher entropy on the visual space, where clusters are more spread but continuous within each class. Aquatic animals, felines, and equines are arranged close to one another but yet separate from other groups, although confusion is still high. In contrast, the space of semantic features is denser, and groups are better formed. We can see the aquatic animals distant from terrestrial ones, and equines and felines forming different groups. Interesting enough, the dog cluster is closer to the cats than the equines.

The combined VSE features, on the other hand, resemble both the visual and semantic space. It has greater entropy than the semantic one, but less than the visual domain, something also observed with the Silhouette Score, that decreases compared to the semantic space, but increases significantly compared to the visual space, showing more cohesion within the samples of the clusters, and separability in between different clusters. We can also observe that the merged space keeps the arrangement of the semantic features, where equines and aquatic animals are neighbors, as well as cats and tigers and leopards, maintaining also the arrangement of these classes shown in the visual space, which indicates that the autoencoder was successful in merging both domains while keeping their main characteristics.

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
VIS	0.1023	0.0605	0.0558	0.0457	0.0336	0.025	0.0233	0.0215	0.0159	0.0149
SEM	0.3049	0.2796	0.1175	0.1038	0.0814	0.0454	0.0266	0.0144	0.0113	0.0087
VSE	0.463	0.2267	0.0844	0.0763	0.0504	0.031	0.0182	0.0089	0.0058	0.0051

Table 2 – PCA variance ratio of the first ten components of the visual (VIS), semantic (SEM), and merged (VSE) spaces for AWA2 dataset, where a subset of twelve classes of animals was chosen: Persian cat, Siamese cat, tiger, leopard, killer whale, dolphin, blue whale, walrus, horse, giraffe, dalmatian, zebra. The first two components of the space formed are displayed in Figure 10.

At last, Table 2 presents the variance ratio of the PCA for the first ten main components of the decomposition presented in Figure 10, where only the first two were displayed. We can observe that the visual space has variances spread among all components, and none of them poses great importance compared to the other. This scenario is different in the semantic space. The first four components have more weight than the rest, so it makes sense to discard them. Meanwhile, the merged space formed by VSE is even simpler. The highest variances are present in the first two main components, which indicates that the feature space displayed in Figure 10 indeed represents well these attributes and that the merge of both domains disentangled the confusion of the third and fourth components in the semantic space.

5.1.1.1 Influence of different semantic features

One of the hypothesis this research studies is that different semantic features provoke different arrangements in the merged space. Figure 11 illustrates how distinctive the latent domain can be depending on the set of semantic features we train the AE with.

Figure 11 – PCA 2D plot of VSE features split into different types: colors, texture, shape, and parts. Note that depending on the set of semantic attributes chosen, the arrangement of the clusters changes in the latent space, reflecting logical concepts such as black and white animals together, animals with spots and patches together, aquatic animals, and animals with four legs together.

Figure 11 shows the results of PCA applied on ResNet50 visual features of Awa2 combined to a specific subset of the semantic features, where the ones chosen keep their original values, while the others we set to zero. Only characteristics extracted from visual representations were selected, so attributes such as smelly, fast, slow, strong, weak, agile, active, and inactive were disregarded. In this context, we separated the features into four groups:

1. **colors:** black, white, blue, brown, gray, orange, red, and yellow;
2. **texture:** patches, spots, stripes, furry, hairless, and tough-skin;
3. **shape:** bulbous, bipedal, quadrupedal, long-leg, and long-neck;
4. **parts:** flippers, hands, paws, tail, horns, and tusks.

As posited, the chosen semantic attributes influences the VSE space, affecting the distribution of similar classes. When using colors: zebras, dalmatians, and killer whales get closer since all these animals are black and white. Tigers, leopards, and giraffes also cluster since they share brown, orange, and yellow colors. On the other hand, the subspace considering texture makes leopards, dalmatians, and giraffes similar, once they all have patches and spots.

Also, zebras and tigers become neighbors since they have stripes. Moreover, in the case of shape and parts, aquatic and terrestrial animals get separated. For parts specially, equines and felines individually form their group, clearly reflecting the semantic space arrangement. For the other cases, even though the semantic features overlap, the visual attributes are distinct. This is an indication that the main features that characterize an animal, semantic, or visual, are used to define the distribution of the class. A direct side effect of this type of procedure is that it allows for better explanation of the feature spaces, and therefore, the results.

Regarding the Silhouette scores, one can observe that they change depending on the set of semantic attributes used, being higher for texture features, but significantly lower than the score of the latent space formed in Figure 10, that uses all attributes. On the other hand, the features related to parts show the lowest score of the four categories analyzed, but it is still higher than the score obtained when using only visual attributes. This result indicates that the semantic data chosen pose great importance to form the latent space.

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
SEM										
COL	0.4374	0.2706	0.1852	0.066	0.0242	0.0144	0.0022	0.0	0.0	0.0
TEX	0.5103	0.2623	0.1661	0.0307	0.022	0.0087	0.0	0.0	0.0	0.0
SHP	0.6831	0.1997	0.0742	0.0229	0.017	0.0031	0.0	0.0	0.0	0.0
PRT	0.5824	0.2771	0.0785	0.0543	0.0077	0.0	0.0	0.0	0.0	0.0
VSE										
COL	0.4546	0.1187	0.0712	0.0546	0.0246	0.0211	0.0178	0.014	0.0129	0.0118
TEX	0.3335	0.2067	0.1107	0.027	0.0251	0.0235	0.0169	0.0164	0.0145	0.0131
SHP	0.4597	0.0802	0.0676	0.037	0.0324	0.0238	0.023	0.0207	0.0168	0.0137
PRT	0.4208	0.1063	0.051	0.0354	0.0318	0.0231	0.021	0.0188	0.0177	0.0166

Table 3 – PCA variance ratio of the first ten components of the semantic (SEM) and merged (VSE) domains for AWA2 dataset. In this case, different sets of semantic attributes were chosen. They represent color (COL), texture (TEX), shape (SHP), and parts (PRT) of twelve classes of animals: Persian cat, Siamese cat, tiger, leopard, killer whale, dolphin, blue whale, walrus, horse, giraffe, dalmatian, zebra. The first two components of the space formed are displayed in Figure 11. The attributes used in each category were listed in Section 5.1.1.1.

Finally, Table 3 shows the PCA variance ratio of the first ten components of the semantic (SEM) and merged (VSE) spaces for AWA2 dataset for each category of features observed in Figure 11. One can observe that for all categories the highest variance is present in the first two or three components of the semantic domain. Similarly, in the merged space, even though the variance is more spread, the first two components still hold most of the variance. Interestingly enough, the parts and shape sub-sets of semantic attributes hold most of the variance in the first two components of the feature space, but have the lowest Silhouette score in the merged space, while texture and colors have a more spread variance among the components, but a better clustering arrangement. This result indicates that the visual features are heavily influencing the results since even though parts and shape properties are better defined, texture and colors are still easier to correlate to the features extracted by the CNN.

5.1.2 Linear Classification

We perform an ablation study where different inputs and components are individually analyzed in order to investigate quantitatively what influences the most on the creation of the latent space. In this sense, VSE is compared against:

- SAE: method developed by [Kodirov, Xiang and Gong \(2017\)](#) for ZSL and clustering. The SVM model has as input the latent space computed using the same configuration the authors set. Since this classification does not work for disjoint datasets, training and test sets were merged and split into 5 stratified folds.
- VIS: the attributes were extracted from the layer immediately before the *softmax* layer of a CNN. For InceptionV1, the input array consists of 1024 features, while for ResNet50 and InceptionV3 this array has 2048 features.
- SEM: the input for SVM classification consists of the semantic attributes annotated by specialists in the case of AwA and CUB datasets, or Amazon’s Mechanical Turk ([VOGEL; SCHIELE, 2004](#)) in the case of SUN and aP&Y databases. Its size varies depending on the dataset evaluated, being 85 for AwA or AwA2, 312 for CUB, 102 for SUN, and 64 or aP&Y.
- CAT: classification based on the evaluation of semantic and visual attributes together. In this case, semantic data is normalized and stacked horizontally at the end of the visual attributes array, forming an input of shape $M \times N$, where M is the number of images and N the number of attributes ($\alpha + \beta$).
- S2S: this classification consists of the combination of SAE and VSE algorithms. SAE is used to define the semantic data, while VSE combines this latent space to the visual one. This approach aims to improve the classification features’ representation while removing the VSE algorithm’s dependency on the semantic array in the test phase.
- PCA: the input for this classification consists of the P most relevant components defined by PCA applied to the CAT data. For direct comparison to SAE and VSE, P has the same length as the length of the semantic array (δ).
- ISO: similar to PCA, the input consists of the P most relevant components defined by Isomap applied to the CAT data, where $P = \delta$, and the number of neighbors $k = 20$.
- LLE: same as ISO but with Locally Linear Embedding (LLE) algorithm applied to the CAT data, where $P = \delta$, and the number of neighbors $k = 20$.

The results for SVM classification can be observed in Table 4. This study performs 5-fold cross-validation to estimate the expected level of fit for each case, so each database is split into

	AwA I1	AwA2 RN	CUB I1	CUB RN	SUN I3	SUN RN	aP&Y I3	aP&Y RN
VIS	85.9 ± 0.4	87.7 ± 0.5	66.5 ± 1.0	64.6 ± 0.3	34.4 ± 0.5	43.4 ± 0.4	64.9 ± 1.0	65.8 ± 0.9
SEM	100 ± 0.0	100 ± 0.0	100 ± 0.0	100 ± 0.0	19.6 ± 0.5	19.2 ± 0.2	72.8 ± 1.3	72.9 ± 1.0
CAT	99.9 ± 0.0	99.7 ± 0.0	99.9 ± 0.0	99.2 ± 0.1	37.6 ± 0.7	45.1 ± 0.8	89.9 ± 0.8	90.5 ± 0.8
SAE	75.0 ± 0.7	75.1 ± 0.8	67.6 ± 0.5	62.4 ± 0.8	3.5 ± 0.5	8.9 ± 1.0	44.7 ± 3.4	50.0 ± 1.8
VSE	100 ± 0.0	100 ± 0.0	99.7 ± 0.0	99.5 ± 0.1	28.6 ± 0.7	37.7 ± 0.5	90.4 ± 0.6	90.4 ± 1.2
S2S	90.4 ± 1.2	83.9 ± 0.4	67.6 ± 0.6	65.8 ± 0.7	27.0 ± 0.3	37.3 ± 0.9	59.6 ± 0.7	61.6 ± 0.9
PCA	100 ± 0.0	100 ± 0.0	100 ± 0.0	99.8 ± 0.1	30.2 ± 0.2	35.4 ± 0.9	91.0 ± 0.5	90.8 ± 1.2
ISO	95.6 ± 0.8	91.9 ± 0.5	71.4 ± 1.6	89.6 ± 0.5	23.9 ± 0.5	27.5 ± 0.7	88.0 ± 0.5	86.6 ± 1.6
LLE	98.5 ± 0.5	97.7 ± 0.0	97.9 ± 0.3	94.1 ± 0.3	27.4 ± 0.7	30.2 ± 0.5	83.6 ± 1.1	80.7 ± 0.3

Table 4 – Results for SVM classification with 0% of semantic degradation. Accuracies are computed using the metric *balanced accuracy score* over 5 stratified folds in an 80 by 20 proportion for determining training and validation data for SAE, VSE, S2S, PCA, ISO, and LLE algorithms. SVM classification is performed over the code created by these algorithms at validation time.

5 stratified folds in an 80 by 20 proportion, and the average result is taken. In the case of VIS, SEM, and CAT algorithms, since no transformation is necessary to generate this data, SVM is trained directly with the training set and validated on the test set. For SAE, VSE, S2S, PCA, ISO, and LLE, the same is done, but the data used for training the SVM is the code generated by each algorithm with the training data as input. Validation is performed over the code generated by the test data. In the case of S2S and VSE, the best model along the 50 training epochs is taken to determine these embedded data. For none of the cases shown the semantic data is disturbed. Hence we expect high levels of accuracy for AwA and CUB since the semantic array for these cases is built upon each class’ characterization.

We use the metric *balanced accuracy score* to compute the classification accuracy considering the distribution of classes. In Table 4 the highest values are highlighted in bold. As expected, the semantic classification for AwA and CUB datasets reach 100%. PCA, CAT, and VSE follow this result. On the other hand, the results for SUN demonstrate a poor structure in the semantic data, so VIS and CAT have the highest accuracy values, being CAT the winner. Finally, aP&Y shows a low accuracy for VIS classification, but CAT, VSE, and PCA could combine the semantic data to it and boost the classification performance.

5.1.2.1 Training History Analysis

In this analysis, we evaluate how VSE performs over its training for each dataset studied. Two metrics are verified: 1) training loss, measured based on the Mean Square Error of the reconstruction of the autoencoder when comparing input and output; and 2) classification accuracy, measured based on SVM *balanced accuracy score* for the codes created at each epoch. Figure 12 shows this training history for both training and validation sets. We can observe that most cases converge after circa 20 epochs, while for SUN, even though training loss converges, classification at validation time does not reach the performance at training time, which indicates a poor generalization and possible overfitting.

Figure 12 – Training loss and classification accuracy along all epochs. Training loss is measured based on the Mean Square Error of the reconstruction of the autoencoder, while classification accuracy corresponds to the results of the *balanced accuracy score* applied over SVM predictions. In the illustration, *i_train* and *i_val* are the training and validation results when using Inception CNN to extract the visual features, while *r_train* and *r_val* are the same but for ResNet CNN.

5.1.2.2 Semantic Degradation

As we can observe from the results summarized in Table 4, the semantic data contributes significantly to the definition of clusters in the latent space, what boosts the class separability. The exception may be the SUN dataset that has a low definition of the semantic data already, curbing its use in the merged space. In this analysis, we aim to observe that phenomenon closely and answer the question: how robust to perturbations in the semantic domain VSE and other embedding algorithms are?

Figure 13 illustrates the results of SVM classification accuracy for all algorithms at eleven different levels of degradation of the semantic data, being 0% the raw data with same results summarized in Table 4, and 100% the semantic array completely replaced by random values. The degradation process follow the steps described in Algorithm 1.

Except for VIS and SAE algorithms that do not rely on the semantic data at the test phase, the semantic test data of the other models is perturbed. We can observe in Figure 13 that for aP&Y and SUN datasets, the concatenation of data is more robust to noise, while CUB and AWA, the manifold algorithms, including VSE, show to be stable for up to 30% of data degradation. As expected, the SEM approach is the one that suffers the most from this random data replacement process since it relies solely on the semantic information.

5.1.3 Zero-Shot Learning

This last evaluation aims to investigate the performance of VSE in a real-world application. Here we compare the method developed against SAE (KODIROV; XIANG; GONG, 2017);

Figure 13 – Results for SVM classification with perturbations in the semantic data at eleven different levels, being 0% classification with no degradation, and 100% classification with all values of the semantic data at test phase replaced by random numbers.

JDL (ZHANG; SALIGRAMA, 2016); SEC, which is an autoencoder coded similarly to VSE, but that uses the approach $f_{zs} : V \rightarrow S$ to define the code; and PCA. The data splits used depend on each database: AWA and CUB use the split recommended by Xian, Schiele and Akata (2017), aP&Y uses as test set the examples in Yahoo database as indicated by Jayaraman and Grauman (2014), and SUN uses the split 707/10, where 10 random classes are chosen to define the test set. This is the same approach used by Kodirov, Xiang and Gong (2017) and recommended by Xiao *et al.* (2010). To allow the reproduction of the results presented, we set the seed of this random class selector to the arbitrary value 1519, which is not necessarily the same one SAE used.

To define each unseen class label, we use the nearest neighbor approach described in Section 4.5.3. SAE and SEC use the function $f_{zs} : V \rightarrow S$ to learn the semantic associations, while VSE, JDL, and PCA use $f_{zs} : V \oplus S \rightarrow K$. The prototypes for the former group are defined as the semantic data itself. For the latter, they are computed from the mean of f_{zs} of three random examples taken for each class. This methodology was chosen to simulate a real-world study since, in this case, not all test examples are available at once, so one must choose a subset of the data to compare each test sample to. The exception is JDL, that has prototypes defined at training time.

Table 4 shows the results for zero-shot learning classification. Similar to Linear Classification, each case was evaluated 5 times, and mean and standard deviation are presented. However, since ZSL has the training and test sets well defined, no cross-validation was performed. Results retrieved from the coded applications are separated from the ones presented in the original articles. It is important to highlight that aP&Y 2008 dataset could not be found in full and that its visual features, as well as SUN visual features, were extracted using Inception V3 (SZEGEDY *et al.*, 2016) instead of Inception V1. That is the reason why SAE^* shows different results from the published ones (KODIROV; XIANG; GONG, 2017). Moreover, the authors do not describe in

	AwA I1	AwA2 RN	CUB I1	CUB RN	SUN I3	SUN RN	aP&Y I3	aP&Y RN
VSE	98.6 ± 0.5	98.5 ± 0.5	92.2 ± 1.0	83.8 ± 1.3	87.4 ± 3.0	87.9 ± 2.3	67.3 ± 3.6	72.7 ± 3.3
PCA	99.8 ± 0.1	99.5 ± 0.2	97.1 ± 0.4	92.8 ± 0.9	87.1 ± 4.5	88.0 ± 3.9	71.9 ± 3.2	76.8 ± 1.9
SEC	54.5 ± 1.4	62.6 ± 1.1	44.4 ± 0.8	42.6 ± 0.3	79.7 ± 1.1	83.3 ± 0.9	12.3 ± 0.5	12.4 ± 0.7
SAE*	84.7 ± 0.0	70.5 ± 0.0	61.4 ± 0.0	55.8 ± 0.0	86.5 ± 0.0	89.0 ± 0.0	12.9 ± 0.0	15.4 ± 0.0
SAE	84.7 ± 0.0		61.4 ± 0.0		91.0 ± 0.0		55.4 ± 0.0	
JDL	80.5 ± 0.5		42.1 ± 0.5		83.8 ± 0.3		50.3 ± 2.9	

Table 5 – Results for ZSL classification using $f_{zs} : V \rightarrow S$ approach (SAE, SAE* and SEC), and $f_{zs} : V \oplus S \rightarrow K$ (VSE, JDL and PCA). SAE and JDL are shown separately because these are the results presented in the original papers, [Kodirov, Xiang and Gong \(2017\)](#) and [Zhang and Saligrama \(2016\)](#) respectively. For the former, only Inception V1 ([SZEGEDY et al., 2015](#)) was used. For the latter, visual attributes were acquired from AlexNet ([KRIZHEVSKY; SUTSKEVER; HINTON, 2012](#)) and VGGNet 19 ([SIMONYAN; ZISSERMAN, 2014](#)). The other results were retrieved from the applications coded in this project.

detail all parameters used for obtaining the results published, hence different values for λ might have been chosen.

From the results, one can observe that PCA has the best values for all cases, which indicates that the linear combination of semantic and visual data might be enough for generalization and for embedding high-level concepts that might be shared across different categories. VSE closely follows these results and outperforms JDL ([ZHANG; SALIGRAMA, 2016](#)). On the other hand, when considering $f_{zs} : V \rightarrow S$ approach, SAE ([KODIROV; XIANG; GONG, 2017](#)) is the best choice.

5.2 Discussion

In this research, we have performed an extensive study on the use of semantic data combined with visual attributes in order to improve class separability in the feature space. As shown in Section 5.1.1, merging high level attributes with features extracted from images embed in the representation abstract concepts that might enhance the distinction of different groups in the latent space, as shown in Figure 10. VSE learned the most relevant features of each domain and built a compact representation that can also reconstruct the original data. In addition, we can see a high level of interpretability of the new domain formed. As expected, different groups of animals formed bigger groups within their own class, such as equines formed by horses, zebras, and giraffes; and spotted/striped felines formed by tigers and leopards. Besides, the cohesion and separability of the clusters improved significantly in the merged domain, enabling the representation of the attributes in a more condensed latent space, where the first two PCA components hold most of the variance of the data.

We have also shown that different sets of semantic attributes result in different arrangements of classes in the feature space, consequently affecting the Silhouette score. For instance, animals with similar shades get closer when only colors are provided, even though they are very distinct. As illustrated in Figure 11, dalmatians, zebras, and killer-whales get together because

they share black and white colorings. The same happens for textures that formed a group of spotty animals such as leopards, dalmatians, and giraffes. Differently, despite the distance of the cluster of dolphins from the aquatic animals in the semantic domain when it comes to shapes, visually these animals still share much in common, so they get closer in the merged domain. This indicates that the main characterization aspects, visual or semantic, are used to define the latent space. These results give the intuition that merging visual and semantic domains may be a good path towards better explaining the results of vision systems based on deep networks. This is also observed when analyzing the variance of the components of the semantic space. Even though the shape and parts attributes present a more linearly defined semantic space, when it comes to the merged domain, texture and colors properties have better-defined clusters, suggesting that these attributes are allegedly easier to correlate with the features extracted from a CNN, what can be concluded presuming that these properties are better outlined in the image domain than shape and parts, and that the computational model is able to obtain a representation that embeds such attributes.

Driven by the idea of better representations in the latent space intuitively, we apply SVM classification to quantify the improvements of the proposed approach. Table 4 summarizes the results for SVM classification for VSE compared against eight other methods. We have seen that when the semantic prototypes are built upon each class' characterization, such as for AwA and CUB, the semantic representation will always reach a perfect classification, followed closely by methods that use this data such as CAT, VSE, and PCA. Moreover, since this semantic definition is linearly separable, it does not require much effort for classification, resulting in a fast convergence, as shown in Figure 12.

On the other hand, the datasets that have the distinct semantic prototypes definition for each image have a lower performance for the semantic representation. However, the combination of visual and semantic domains still boost the classification. In the case of aP&Y, the merge was able to create a space linearly separable enough to allow PCA to perform well on it, followed by VSE and CAT. Differently, in the case of SUN the best results were obtained for the direct concatenation of both data. This means that the visual and semantic domain cannot be combined because it leads to poor generalization, as evidenced in Figure 12. This result might be a consequence of the dataset's high granularity, which has 717 different classes of scenes that share common attributes, hence the low performance on the semantic domain. Moreover, ImageNet was not originally built for scene classification. Since the visual features were extracted from CNNs trained over it, the visual representations will not be accurate.

As said previously, manually annotating data is costly and impractical. Therefore, when proposing the use of the semantic and visual data together, we must account for not creating a bottleneck in this application. In this sense, automatic extractors are a good alternative but are susceptible to errors. Hence, the application that uses this approach must be robust to noise up to some level. In Section 5.1.2.2, we investigated the performance of the methods studied when the

semantic data is perturbed.

The results have shown that, except for SUN, VSE is robust for up 20% of degradation, which is an important finding considering that most real-world applications will hardly accept data with more than this amount of noise, so our proposal proves to be viable in this sense. In the case of AwA and CUB, LLE goes beyond 20%, but the subspace formed cannot reconstruct the original data, something that VSE can do. Contrarily, SUN and APY have better results for CAT data. These are the datasets where semantic prototypes are defined upon each image based on Amazon’s Mechanical Turk (VOGEL; SCHIELE, 2004) tool. We can see from Table 4 that the semantic data is not perfectly well defined, mainly in the case of SUN, so it is expected that a combination of this data to the visual one will not help much in constructing a better feature space. More than that, discarding visual data might be harmful. That is what is shown in Figure 13, SUN needs all visual features to keep the classification accuracy up to some level. Finally, looking into PCA we see that it keeps a good accuracy level up to some perturbation levels, indicating that space formed still has some linearity. However, when the degradation increases too much, non-linear methods are better.

Finally, the last investigation includes the analysis of the performance of VSE for zero-shot learning. As shown in Table 5, VSE was compared against SAE, SEC, JDL, and PCA. From the results, we can see that, oddly, PCA has a better performance in most cases. PCA is a method for dimensionality reduction that uses linear combinations to project the data into a centered and orthogonal feature space, where the component with the highest variability is the most important one. The values shown in Table 5 indicate that the linear combination of the semantic and visual domain is enough to learn the high level features shared among seen and unseen classes that enable zero-shot classification, which would mean that the highest variance is in the semantic features, but is something that can only be confirmed with more analysis. Following PCA, VSE is the second most efficient method, outperforming JDL. This indicates that the joint embeddings formed by PCA and VSE are more robust than the joint discriminative learning framework proposed by Zhang and Saligrama (2016), suggesting that $f_{zs} : V \oplus S \rightarrow K$ can be more efficient than $f_{zs} : V \rightarrow S$ when the comparison of K to the prototypes is direct. This is different from what Zhang and Saligrama (2016) do, which consists in embedding only the target domain at test time and compare it to the source embeddings created at training time, removing the need for semantic data at test time. Contrarily, PCA and VSE need both data at test time and still fall under the concern that semantic and visual features may not be always available. Hence, robustness to semantic degradation must still be evaluated.

CONCLUSION

In this project, we studied the influence of semantic attributes in the construction of new representations of RGB images. For that, we analyze the efficiency of the application of an autoencoder to combine visual and semantic data, reducing the dimensionality of the array of characteristics by selecting the most representative properties of an image, and consequently abstracting in the process high-level concepts of each group. With that, we aimed to create a space of features more distinctive among classes and learn representations that transcend class boundaries, allowing us to improve the performance of image classification and clustering applications and apply zero-shot learning.

To perform the experiments, we studied four benchmarks used for zero-shot learning: Animals with Attributes 1 and 2, CUB200-2011, aPascal&Yahoo, and SUN Attributes, where semantic aspects were defined either by specialists or by Amazon’s Mechanical Turk tool; and use three different convolutional neural networks to extract the visual properties: Inception V1, Inception V3 and ResNet 50. The method developed is called Visual-Semantic Encoder (VSE) and was compared against the state-of-the-art algorithm for zero-shot learning SAE (KODIROV; XIANG; GONG, 2017), and other solid methods used for dimensionality reduction such as PCA, Isomap, and Locally Linear Embedding.

To investigate the influence of different components and the robustness of the algorithm proposed, we perform an ablation study where we use distinct combinations of the visual and semantic input and introduce noise to the semantic array at different degrees of perturbation. In addition, for zero-shot learning we compare the use of the proposed neural network autoencoder in two distinct learning frameworks: learning a projection from visual to the semantic domain ($f_{zs} : V \rightarrow S$); and learning the embedding of visual and semantic data into a single representation ($f_{zs} : V \oplus S \rightarrow K$).

As a result, our visual-semantic space was shown to be discriminative for both coarse and fine-grained classification tasks. The VSE learns the most relevant features of each domain,

building a compact representation that can also reconstruct the original data. The semantic data enriched the visual representation of concepts in the feature space and inherited the high-level concepts of each class. By combining the advantages of visual and semantic features, we also allow for interpreting the class distributions by investigating, for example, the influence of texture, color, and shape attributes. We believe this may be a good path towards better explaining the results of vision systems based on deep networks. Our experiments also suggest that the addition of semantic data into the representation of an image introduces linearity in the feature space, allowing the use of simple dimensionality reduction methods to combine the most representative features such as PCA. However, for fine-grained and noisy data, non-linear methods are better. In addition, when the semantic features are not reliable, the combination of visual and semantic domains into a lower dimension representation is inefficient, being the simple concatenation the attributes of both domain more adequate.

In respect to zero-shot learning, we have seen that frameworks that use the combination of visual and semantic data are more efficient when the comparison of the embedding formed with the prototypes is direct, implying that the features of the merged representation contain data that transcend class boundaries, which allows the learning of high-level concepts that permit identifying unknown objects. Nevertheless, this approach needs to create prototypes and depends on the semantic data at the test phase, which may require automatic extractors in real-world applications, so robustness to noise must yet be investigated.

In conclusion, we have shown that the use of semantic data in an image embedding improves its discrimination and interpretation. This can be done efficiently by an autoencoder, that can cope up to 20% of perturbation on semantic data. This robustness would allow to obtain semantic features from off-the-shelf models such as RCNNs (ZHANG *et al.*, 2018) or object detection networks (REDMON; FARHADI, 2018), since even imprecise semantic data would still improve results. This approach also has the advantage of compacting the representation of the feature space, while allowing the reconstruction of the original data, mitigating this way the curse-of-dimensionality. For future works, we recommend studying the feature space created by the merge to better understand its linear properties. Additionally, we suggest the verification of robustness to noise in the combination of data for zero-shot learning applications. At last, it might be a good alternative for image retrieval applications to learn features from the semantic to the visual domain ($f_{zs} : S \rightarrow V$), which might allow the comparison of images through high-level concepts, a task known as disentanglement. For instance, one could retrieve round objects images by locating the visual features linked to this property via an autoencoder. All those demonstrate the relevance of our study and its usefulness for future research.

BIBLIOGRAPHY

ABDI, H.; WILLIAMS, L. J. Principal component analysis. **Wiley interdisciplinary reviews: computational statistics**, Wiley Online Library, v. 2, n. 4, p. 433–459, 2010. Citation on page 33.

AKATA, Z.; PERRONNIN, F.; HARCHAOUI, Z.; SCHMID, C. Label-embedding for image classification. **IEEE transactions on pattern analysis and machine intelligence**, IEEE, v. 38, n. 7, p. 1425–1438, 2015. Citations on pages 26, 27, and 44.

ALMOUSLI, H.; VINCENT, P. Semi supervised autoencoders: better focusing model capacity during feature extraction. In: SPRINGER. **International Conference on Neural Information Processing**. [S.l.], 2013. p. 328–335. Citations on pages 22, 32, and 41.

BALDI, P. Autoencoders, unsupervised learning, and deep architectures. In: **Proceedings of ICML workshop on unsupervised and transfer learning**. [S.l.: s.n.], 2012. p. 37–49. Citations on pages 31 and 32.

BARTELS, R. H.; STEWART, G. W. Solution of the matrix equation $ax + xb = c$ [f4]. **Communications of the ACM**, ACM New York, NY, USA, v. 15, n. 9, p. 820–826, 1972. Citations on pages 32 and 39.

BERNSTEIN, M.; SILVA, V. D.; LANGFORD, J. C.; TENENBAUM, J. B. **Graph approximations to geodesics on embedded manifolds**. [S.l.], 2000. Citation on page 34.

BEZDEK, J. C.; KELLER, J.; KRISNAPURAM, R.; PAL, N. **Fuzzy models and algorithms for pattern recognition and image processing**. [S.l.]: Springer Science & Business Media, 1999. Citation on page 21.

BIEDERMAN, I. Recognition-by-components: a theory of human image understanding. **Psychological review**, American Psychological Association, v. 94, n. 2, p. 115, 1987. Citation on page 21.

BRODERSEN, K. H.; ONG, C. S.; STEPHAN, K. E.; BUHMANN, J. M. The balanced accuracy and its posterior distribution. In: IEEE. **2010 20th International Conference on Pattern Recognition**. [S.l.], 2010. p. 3121–3124. Citation on page 50.

CAVALLARI, G.; RIBEIRO, L.; PONTI, M. Unsupervised representation learning using convolutional and stacked auto-encoders: a domain and cross-domain feature space analysis. In: IEEE. **2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)**. [S.l.], 2018. p. 440–446. Citation on page 32.

CAYTON, L. Algorithms for manifold learning. **Univ. of California at San Diego Tech. Rep**, v. 12, n. 1-17, p. 1, 2005. Citations on pages 22, 33, 34, and 41.

CHOLLET, F. **Keras**. [S.l.]: GitHub, 2015. <<https://github.com/fchollet/keras>>. Citation on page 46.

CORTES, C.; VAPNIK, V. Support-vector networks. **Machine learning**, Springer, v. 20, n. 3, p. 273–297, 1995. Citation on page 49.

COX, M. A.; COX, T. F. Multidimensional scaling. In: **Handbook of data visualization**. [S.l.]: Springer, 2008. p. 315–347. Citation on page 34.

DENG, J.; DONG, W.; SOCHER, R.; LI, L.-J.; LI, K.; FEI-FEI, L. ImageNet: A Large-Scale Hierarchical Image Database. In: **CVPR09**. [S.l.: s.n.], 2009. Citations on pages 21, 28, and 46.

DESELAERS, T.; FERRARI, V. Visual and semantic similarity in imagenet. In: IEEE. **CVPR 2011**. [S.l.], 2011. p. 1777–1784. Citation on page 37.

EVERINGHAM, M.; GOOL, L. V.; WILLIAMS, C. K.; WINN, J.; ZISSERMAN, A. **The PASCAL Visual Object Classes Challenge 2008 (VOC2008) results**. 2008. Citation on page 47.

FARHADI, A.; ENDRES, I.; HOIEM, D.; FORSYTH, D. Describing objects by their attributes. In: IEEE. **2009 IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.], 2009. p. 1778–1785. Citations on pages 21, 22, 25, 26, 35, 37, 44, 46, and 47.

FERREIRA, M. D.; CORRÊA, D. C.; NONATO, L. G.; MELLO, R. F. de. Designing architectures of convolutional neural networks to solve practical problems. **Expert Systems with Applications**, Elsevier, v. 94, p. 205–217, 2018. Citation on page 30.

FRIEDMAN, J. H. On bias, variance, 0/1—loss, and the curse-of-dimensionality. **Data mining and knowledge discovery**, Springer, v. 1, n. 1, p. 55–77, 1997. Citations on pages 22 and 41.

FU, Y.; HOSPEDALES, T. M.; XIANG, T.; GONG, S. Transductive multi-view zero-shot learning. **IEEE transactions on pattern analysis and machine intelligence**, IEEE, v. 37, n. 11, p. 2332–2345, 2015. Citation on page 35.

GE, Z.; DEMYANOV, S.; BOZORGTABAR, B.; ABEDINI, M.; CHAKRAVORTY, R.; BOWLING, A.; GARNAVI, R. Exploiting local and generic features for accurate skin lesions classification using clinical and dermoscopy imaging. In: IEEE. **2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)**. [S.l.], 2017. p. 986–990. Citation on page 43.

GIRSHICK, R.; DONAHUE, J.; DARRELL, T.; MALIK, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2014. p. 580–587. Citation on page 26.

GONZALES, R. C.; WOODS, R. E. **Digital image processing**. [S.l.]: Prentice hall New Jersey, 2002. Citations on pages 15, 27, and 28.

GONZALEZ, R. C.; THOMASON, M. G. Syntactic pattern recognition: An introduction. Addison-Wesley Publishing Company, Reading, MA, 1978. Citation on page 21.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A.; BENGIO, Y. **Deep learning**. [S.l.]: MIT press Cambridge, 2016. Citations on pages 15, 25, 28, 29, 30, 32, and 71.

HARDOON, D. R.; SZEDMAK, S.; SHAWE-TAYLOR, J. Canonical correlation analysis: An overview with application to learning methods. **Neural computation**, MIT Press, v. 16, n. 12, p. 2639–2664, 2004. Citation on page 38.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. **The elements of statistical learnin**. [S.l.]: New York: Springer, 2009. Citations on pages [31](#) and [33](#).

HE, K.; SUN, J. Convolutional neural networks at constrained time cost. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2015. p. 5353–5360. Citation on page [44](#).

HE, K.; ZHANG, X.; REN, S.; SUN, J. Deep residual learning for image recognition. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2016. p. 770–778. Citations on pages [21](#), [22](#), [28](#), [43](#), and [44](#).

HINTON, G. E.; SALAKHUTDINOV, R. R. Reducing the dimensionality of data with neural networks. **science**, American Association for the Advancement of Science, v. 313, n. 5786, p. 504–507, 2006. Citation on page [32](#).

HOTELLING, H. Relations between two sets of variates. In: **Breakthroughs in statistics**. [S.l.]: Springer, 1992. p. 162–190. Citation on page [38](#).

JAYARAMAN, D.; GRAUMAN, K. Zero-shot recognition with unreliable attributes. In: **Advances in neural information processing systems**. [S.l.: s.n.], 2014. p. 3464–3472. Citations on pages [26](#) and [58](#).

JUAN, D.-C.; LU, C.-T.; LI, Z.; PENG, F.; TIMOFEEV, A.; CHEN, Y.-T.; GAO, Y.; DUERIG, T.; TOMKINS, A.; RAVI, S. Graph-rise: Graph-regularized image semantic embedding. **arXiv preprint arXiv:1902.10814**, 2019. Citations on pages [27](#) and [37](#).

KODIROV, E.; XIANG, T.; GONG, S. Semantic autoencoder for zero-shot learning. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2017. p. 3174–3183. Citations on pages [11](#), [13](#), [15](#), [17](#), [22](#), [23](#), [27](#), [32](#), [34](#), [35](#), [36](#), [38](#), [39](#), [42](#), [45](#), [46](#), [49](#), [50](#), [55](#), [57](#), [58](#), [59](#), and [63](#).

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: **Advances in neural information processing systems**. [S.l.: s.n.], 2012. p. 1097–1105. Citations on pages [17](#), [27](#), and [59](#).

LAMPERT, C. H.; NICKISCH, H.; HARMELING, S. Learning to detect unseen object classes by between-class attribute transfer. In: **IEEE. 2009 IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.], 2009. p. 951–958. Citations on pages [21](#), [34](#), [35](#), and [37](#).

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **nature**, Nature Publishing Group, v. 521, n. 7553, p. 436, 2015. Citation on page [30](#).

LIN, T.-Y.; ROYCHOWDHURY, A.; MAJI, S. Bilinear cnn models for fine-grained visual recognition. In: **Proceedings of the IEEE International Conference on Computer Vision**. [S.l.: s.n.], 2015. p. 1449–1457. Citation on page [37](#).

LU, Y. Unsupervised learning on neural network outputs: with application in zero-shot learning. **arXiv preprint arXiv:1506.00990**, 2015. Citation on page [38](#).

MELLO, R. F.; PONTI, M. A. **Machine Learning: A Practical Approach on the Statistical Learning Theory**. [S.l.]: Springer, 2018. Citations on pages [22](#), [25](#), [30](#), [33](#), and [49](#).

MESCHEDER, L.; NOWOZIN, S.; GEIGER, A. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. **arXiv preprint arXiv:1701.04722**, 2017. Citation on page 32.

MICHALSKI, R. S.; CARBONELL, J. G.; MITCHELL, T. M. **Machine learning: An artificial intelligence approach**. [S.l.]: Springer Science & Business Media, 2013. Citation on page 25.

MIKOLOV, T.; CHEN, K.; CORRADO, G.; DEAN, J. Efficient estimation of word representations in vector space. **arXiv preprint arXiv:1301.3781**, 2013. Citations on pages 26 and 42.

MIKOLOV, T.; SUTSKEVER, I.; CHEN, K.; CORRADO, G. S.; DEAN, J. Distributed representations of words and phrases and their compositionality. In: **Advances in neural information processing systems**. [S.l.: s.n.], 2013. p. 3111–3119. Citations on pages 26 and 42.

MORITA, S. **Geometry of differential forms**. [S.l.]: American Mathematical Soc., 2001. Citations on pages 31 and 71.

NAIR, V.; HINTON, G. E. Rectified linear units improve restricted boltzmann machines. In: **ICML**. [S.l.: s.n.], 2010. Citations on pages 43, 44, and 46.

NSIMBA, C. B.; LEVADA, A. L. Nonlinear dimensionality reduction in texture classification: Is manifold learning better than pca? In: SPRINGER. **International Conference on Computational Science**. [S.l.], 2019. p. 191–206. Citation on page 33.

PATTERSON, G.; HAYS, J. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In: IEEE. **2012 IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.], 2012. p. 2751–2758. Citations on pages 22, 27, 45, and 47.

PATTERSON, G.; XU, C.; SU, H.; HAYS, J. The sun attribute database: Beyond categories for deeper scene understanding. **International Journal of Computer Vision**, Springer, v. 108, n. 1-2, p. 59–81, 2014. Citations on pages 27 and 46.

PONTI, M. A.; COSTA, G. B. P. da. Como funciona o deep learning. **arXiv preprint arXiv:1806.07908**, 2018. Citation on page 30.

PONTI, M. A.; RIBEIRO, L. S. F.; NAZARE, T. S.; BUI, T.; COLLOMOSSE, J. Everything you wanted to know about deep learning for computer vision but were afraid to ask. In: IEEE. **2017 30th SIBGRAPI conference on graphics, patterns and images tutorials (SIBGRAPI-T)**. [S.l.], 2017. p. 17–41. Citations on pages 29 and 30.

RANZATO, M.; BOUREAU, Y.-L.; CHOPRA, S.; LECUN, Y. A unified energy-based framework for unsupervised learning. In: **Artificial Intelligence and Statistics**. [S.l.: s.n.], 2007. p. 371–379. Citations on pages 22, 31, 32, and 41.

REDMON, J.; FARHADI, A. Yolov3: An incremental improvement. **arXiv preprint arXiv:1804.02767**, 2018. Citations on pages 26, 42, and 64.

REN, Z.; JIN, H.; LIN, Z.; FANG, C.; YUILLE, A. Multi-instance visual-semantic embedding. **arXiv preprint arXiv:1512.06963**, 2015. Citations on pages 27 and 37.

RIAZA, R. **Differential-algebraic systems: analytical aspects and circuit applications**. [S.l.]: World Scientific, 2008. Citations on pages 31 and 71.

ROSCH, E. Cognitive representations of semantic categories. **Journal of experimental psychology: General**, American Psychological Association, v. 104, n. 3, p. 192, 1975. Citation on page 25.

_____. Principles of categorization. **Concepts: core readings**, v. 189, 1999. Citation on page 25.

ROUSSEEUW, P. J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. **Journal of computational and applied mathematics**, Elsevier, v. 20, p. 53–65, 1987. Citation on page 49.

ROWEIS, S. T.; SAUL, L. K. Nonlinear dimensionality reduction by locally linear embedding. **science**, American Association for the Advancement of Science, v. 290, n. 5500, p. 2323–2326, 2000. Citation on page 34.

RUMELHART, D.; HINTON, G.; WILLIAMS, R. Learning internal representation by error propagation, parallel distributed processing. **MIT Press, Cambridge**, 1986. Citation on page 31.

RUSS, J. C. Image processing. In: **Computer-assisted microscopy**. [S.l.]: Springer, 1990. p. 33–69. Citation on page 27.

SAUL, L. K.; ROWEIS, S. T. An introduction to locally linear embedding. **unpublished. Available at: <http://www.cs.toronto.edu/~roweis/lle/publications.html>**, 2000. Citation on page 34.

SEWAK, M.; KARIM, M. R.; PUJARI, P. **Practical Convolutional Neural Networks: Implement Advanced Deep Learning Models Using Python**. [S.l.]: Packt Publishing Ltd, 2018. Citations on pages 28 and 29.

SILBERER, C.; FERRARI, V.; LAPATA, M. Models of semantic representation with visual attributes. In: **Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**. [S.l.: s.n.], 2013. v. 1, p. 572–582. Citation on page 21.

SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. **arXiv preprint arXiv:1409.1556**, 2014. Citations on pages 17 and 59.

SRIVASTAVA, R. K.; GREFF, K.; SCHMIDHUBER, J. Highway networks. **arXiv preprint arXiv:1505.00387**, 2015. Citation on page 44.

SU, Y.; JURIE, F. Improving image classification using semantic attributes. **International journal of computer vision**, Springer, v. 100, n. 1, p. 59–77, 2012. Citations on pages 26 and 37.

SZEGEDY, C.; LIU, W.; JIA, Y.; SERMANET, P.; REED, S.; ANGUELOV, D.; ERHAN, D.; VANHOUCHE, V.; RABINOVICH, A. Going deeper with convolutions. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2015. p. 1–9. Citations on pages 17, 21, 22, 28, 38, 43, and 59.

SZEGEDY, C.; VANHOUCHE, V.; IOFFE, S.; SHLENS, J.; WOJNA, Z. Rethinking the inception architecture for computer vision. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2016. p. 2818–2826. Citations on pages 22, 43, 44, and 58.

VOGEL, J.; SCHIELE, B. Natural scene retrieval based on a semantic modeling step. In: SPRINGER. **International Conference on Image and Video Retrieval**. [S.l.], 2004. p. 207–215. Citations on pages [45](#), [47](#), [55](#), and [61](#).

WAH, C.; BRANSON, S.; WELINDER, P.; PERONA, P.; BELONGIE, S. **The Caltech-UCSD Birds-200-2011 Dataset**. [S.l.], 2011. Citations on pages [22](#), [44](#), and [46](#).

WANG, J.; YANG, Y.; MAO, J.; HUANG, Z.; HUANG, C.; XU, W. Cnn-rnn: A unified framework for multi-label image classification. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2016. p. 2285–2294. Citations on pages [27](#) and [37](#).

XIAN, Y.; AKATA, Z.; SHARMA, G.; NGUYEN, Q.; HEIN, M.; SCHIELE, B. Latent embeddings for zero-shot classification. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2016. p. 69–77. Citations on pages [27](#) and [44](#).

XIAN, Y.; LAMPERT, C. H.; SCHIELE, B.; AKATA, Z. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. **IEEE transactions on pattern analysis and machine intelligence**, IEEE, v. 41, n. 9, p. 2251–2265, 2018. Citations on pages [22](#), [44](#), and [46](#).

XIAN, Y.; LORENZ, T.; SCHIELE, B.; AKATA, Z. Feature generating networks for zero-shot learning. In: **The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2018. Citation on page [43](#).

XIAN, Y.; SCHIELE, B.; AKATA, Z. Zero-shot learning - the good, the bad and the ugly. In: **IEEE Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2017. Citation on page [58](#).

XIAO, J.; HAYS, J.; EHINGER, K. A.; OLIVA, A.; TORRALBA, A. Sun database: Large-scale scene recognition from abbey to zoo. In: IEEE. **2010 IEEE computer society conference on computer vision and pattern recognition**. [S.l.], 2010. p. 3485–3492. Citations on pages [47](#) and [58](#).

XU, H.; QI, G.; LI, J.; WANG, M.; XU, K.; GAO, H. Fine-grained image classification by visual-semantic embedding. In: **IJCAI**. [S.l.: s.n.], 2018. p. 1043–1049. Citations on pages [21](#), [34](#), [35](#), [37](#), and [47](#).

YOSINSKI, J.; CLUNE, J.; BENGIO, Y.; LIPSON, H. How transferable are features in deep neural networks? In: **Advances in neural information processing systems**. [S.l.: s.n.], 2014. p. 3320–3328. Citation on page [43](#).

ZHANG, J.; WU, Q.; SHEN, C.; ZHANG, J.; LU, J. Multilabel image classification with regional latent semantic dependencies. **IEEE Transactions on Multimedia**, IEEE, v. 20, n. 10, p. 2801–2813, 2018. Citations on pages [26](#), [27](#), [37](#), [42](#), and [64](#).

ZHANG, Z.; SALIGRAMA, V. Zero-shot learning via joint latent similarity embedding. In: **proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2016. p. 6034–6042. Citations on pages [11](#), [13](#), [15](#), [17](#), [23](#), [27](#), [38](#), [39](#), [40](#), [58](#), [59](#), and [61](#).

GLOSSARY

Autoencoder: model composed of three structures: a code z , defined as the latent space between input x and output $\hat{x}$; an encoder A , responsible for encoding the input by predicting the best values for the latent code z ; and a decoder B , that reconstructs from the code z the output $\hat{x}$.

Classification: a task in machine learning based on feature learning where the computer program is asked to specify on which of k categories some input belongs to ([GOODFELLOW *et al.*, 2016](#)). When $k > 1$ this task is called multi-label classification.

Feature: also called attribute, it is a quantitative representation of a specific characteristic of an object.

Latent space: an space resultant from a projection that is more representative than the feature space (source domain).

Manifold: a topological space where each point has its neighborhood locally homeomorphic to an subset of an Euclidean space ([RIAZA, 2008](#); [MORITA, 2001](#)).

Semantic features: high level descriptions that characterize an object formed by the mapping of attributes elucidated by humans to describe a particular class.

Unsupervised Learning: type of learning where the output y is not available at training time, hence cannot guide the learning process. This approach is usually given by the association or clustering rules that estimate the probability of a given instance by evaluating how the space of features is organized.

Variance learning: analysis of the feature space in such a way that once the features are projected, the organization in space of a given point and its neighbors should be preserved from the source domain to the target domain.

Visual features: quantitative values related to size, shape, position, density, color, and texture, which can help a machine recognize, count, or measure objects within an image or video.

Zero-shot learning: type of classification where the goal is to recognize unseen classes at test level based on shared high level concepts learned at training level.