

Aprendendo representações com Deep Learning: arquiteturas, treinamento e transferência de aprendizado

Fernando Pereira dos Santos

fernando_persan@alumni.usp.br
fernando.santos@birdie.ai
<https://www.linkedin.com/in/fernando-persan/>

October, 2020



Topics

- **Transfer learning concepts**
- Feature extraction using pre-trained networks
- Fine-tuning network
- Unsupervised and semi-supervised approaches

<https://github.com/fernandopersan/4EscolaBigData>

Pattern Recognition Tasks

- **Instances:** each example that constitutes a database;
- **Feature spaces:** a set of attributes that describes each instance;
- **Learning Task:** involves absorbing instances and adjusting descriptors to provide a feature space, which should be descriptive enough for all provided data.

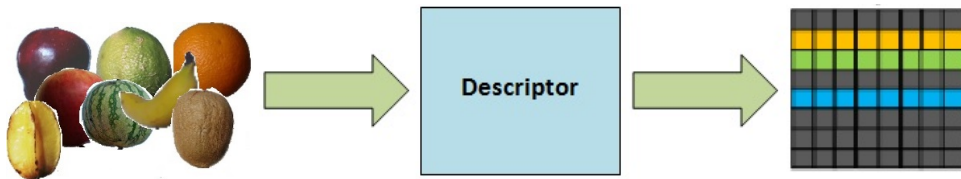


Figure 1: Instances → Descriptor → Feature Space.

Pattern Recognition Tasks

$$f : X \longrightarrow Y, \text{ where } \begin{cases} f : \text{prediction function} \\ X : \text{instances} \\ Y : \text{labels} \end{cases} \quad (1)$$

Predictive models assume that source $\{X_s, Y_s\}$ and target $\{X_t, Y_t\}$ sets share the same data distribution:

- **source** (used to infer the model) and **target** sets (any unseen data);
- When the **target set differs from the source training set**, one may need to consider to fully reconstruct the original model from scratch, retraining it with new data, which **may be expensive and sometimes impossible**;
- The possibility of **reusing similar and large datasets** would guarantee the reduction in effort to recollect new data.

Pattern Recognition Tasks

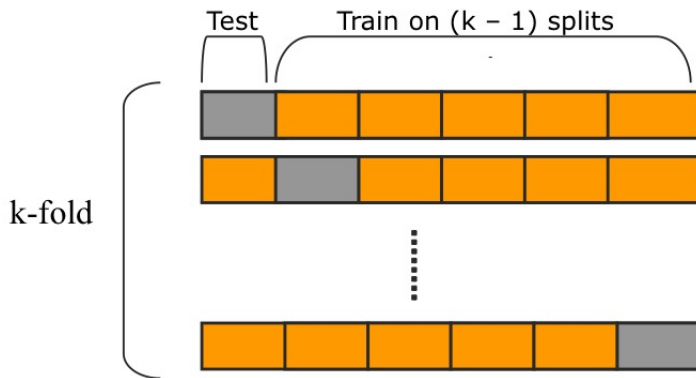


Figure 2: Cross-validation. Source: <https://gusrabbit.com/assets/images/kfold.jpg>.

Transfer Learning definition

Definition:

“Transfer learning is the improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned” [Torrey and Shavlik, 2010].

Definition:

“Transfer learning and domain adaptation refer to the situation where what has been learned in one setting can be exploited to improve generalization in another setting” [Goodfellow et al., 2016].

Transfer Learning definition

$$D = \{X, P(X)\}, \text{ where } \begin{cases} X = \{x_1, \dots, x_n\} : \text{feature space} \\ P(X) : \text{probabilistic distribution} \end{cases} \quad (2)$$

$$T = \{Y, f(\cdot)\}, \text{ where } \begin{cases} Y = \{y_1, \dots, y_m\} : \text{label space} \\ f(\cdot) : \text{prediction function} \end{cases} \quad (3)$$

$f(\cdot)$ models $P(y|x)$ for all $y \in Y$ and $x \in X$

Given a source domain D_s and a learning task T_s , a target domain D_t and a learning task T_t , TL aims to improve the function learning of the target prediction in D_t using the knowledge in D_s and T_s , where $D_s \neq D_t$ **or** $T_s \neq T_t$.

Scenarios of Transfer Learning

Inductive TL occurs when there are **differences in the task** it proposes to perform from the previously learned one, in which **some of the labeled data from the target domain is required** to adapt the knowledge from the source domain.

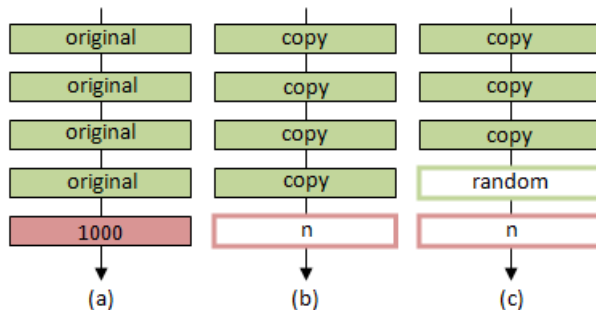


Figure 3: Example of Inductive Transfer Learning: fine-tuning network.

Scenarios of Transfer Learning

Transductive TL **does not require target domain information** during the transfer task, only the **availability of source domain labels**, which the source and target tasks are similar.

****Domain Adaptation:** it occurs when the feature spaces are similar (source and target) and the marginal probability distributions are different.

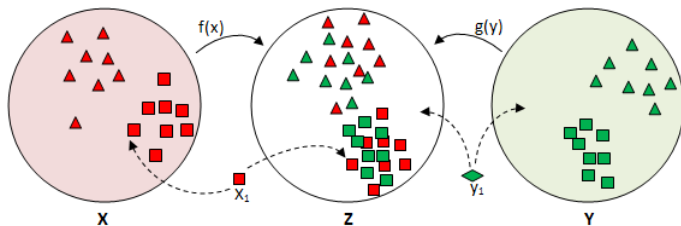


Figure 4: Example of Transductive Transfer Learning: feature space alignment.

Feature Transfer Learning

Feature TL: a particular case, where the dissimilarity between data distributions from the source feature space in relation to the target feature space is minimized, regardless of the feature extraction method used.

If the source and the target datasets are sufficiently similar, it is expected that the model has acceptable performance between those datasets. The main challenge in TL is to correlate the source training data distribution to the target test data distribution:

- **what to transfer** by investigating the similarity between domains in which common peculiarities must be highlighted and discrepancies must be minimized (supervised and/or unsupervised models, learned features, and parameters);
- **how to transfer** the knowledge (exploring machine learning techniques and pattern recognition);
- **when to transfer** detecting scenarios where transfer is useful and avoiding negative transfer.

Negative transfer: occurs when the acquired knowledge worsens the model performance.

Meaning of Feature Transfer Learning for Classification Tasks

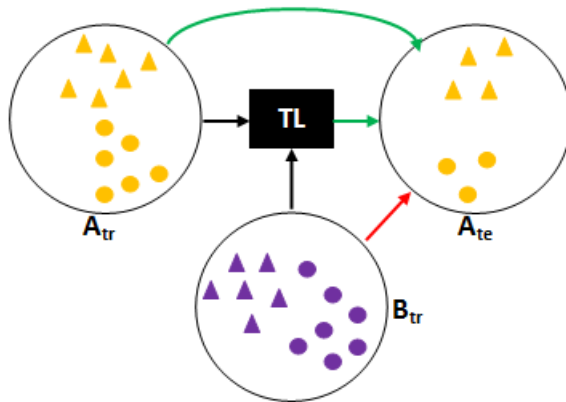


Figure 5: Expected scenario in Feature TL for classification tasks.

Meaning of Feature Transfer Learning for Anomaly Detection Tasks

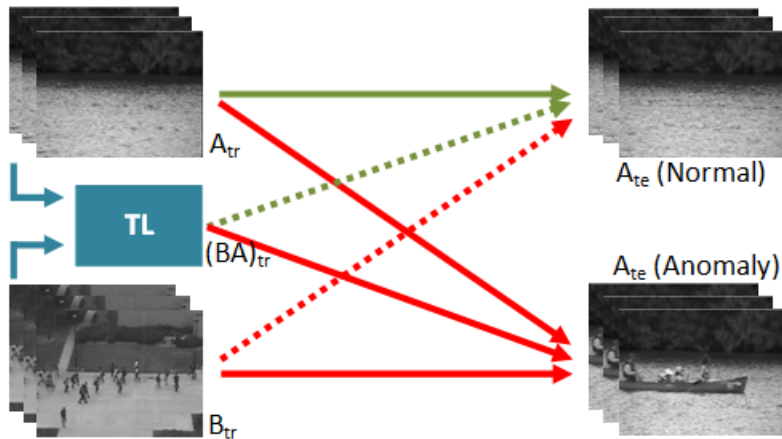


Figure 6: Expected scenario in Feature TL for anomaly detection tasks [dos Santos et al., 2019].

Generalization

Generalization:

Generalization is a divergence measure of how well a classifier or detector performance with unseen data (test) is consistent with its performance on seen data (training).

Generalization:

- one of reasons for a model not to be completely adaptable to several domains is the absence of evaluation by generalization measures;
- often, models are evaluated only by classical measures of the assigned task;
- the behavior of one model in other datasets is rarely evaluated.

A good measure of domain generalization can indicate which source dataset is more suitable to a target dataset or demonstrate which model is more able to transfer the learning for a given scenario.

How can one measure generalization of a feature space produced by some method?

Cross-Domain Feature Space Generalization (CDFG) Measure

CDFG Measure is based on concepts from **Statistical Learning Theory**, expressed as:

$$|R_{emp}(f_n) - R(f_n)|, \quad (4)$$

- $R_{emp}(f_n)$ is the risk of a classifier f_n evaluated over the training set (the empirical risk)
- $R(f_n)$ is the true risk (expectancy of loss) of same f_n over “all data” (called expected risk)

Partial Cross-domain Feature Space Generalization (G_{part}):

$$G_{part}(f_n^A) = |R(f_n^A)_{x \in \mathcal{X}^A} - R(f_n^A)_{x \in \mathcal{X}^B}| \quad (5)$$

Complete Cross-domain Feature Space Generalization (G_{comp}):

$$G_{comp}(f_n^A, f_n^B) = \frac{1}{2} \left(|R(f_n^A)_{x \in \mathcal{X}^A} - R(f_n^A)_{x \in \mathcal{X}^B}| + |R(f_n^B)_{x \in \mathcal{X}^B} - R(f_n^B)_{x \in \mathcal{X}^A}| \right) \quad (6)$$

Cross-Domain Feature Space Generalization (CDFG) Measure

Restrictions:

- the set of admissible functions from the classifier/detector are the same, e.g. same parameters on SVM training setup;
- both feature spaces \mathcal{X}^A and \mathcal{X}^B are described by the same set of descriptors;
- the domain mapping method has no prior knowledge of unseen (test) data on either domain.

$$G_{part}(f_{\alpha}^A) < G_{part}(f_{\beta}^A) \quad (7)$$

$$G_{part}(f_{\alpha}^B) < G_{part}(f_{\beta}^B) \quad (8)$$

$$G_{comp}(f_{\alpha}^A, f_{\alpha}^B) < G_{comp}(f_{\beta}^A, f_{\beta}^B) \quad (9)$$

CDFG Measure on Anomaly Detection evaluation (Example)



Figure 7: Samples of test frames from: Ped1; Ped2; Bellevue; and Train. Anomalous events are represented in red (trucks, cyclists, vehicle conversions, and passenger movement). Examples of normal events are in green (pedestrians and straight-line pass). Adapted from [dos Santos et al., 2019].

CDFG Measure on Anomaly Detection evaluation (Example)

Table 1: Anomaly detection in urban scenarios (EER: Equal Error Rate) [dos Santos et al., 2019].

Source → Target	Cross-feature	PCA	TCA
Ped1 → Ped1	51.4	35.17	39.68
Ped2 → Ped1	51.46	39.13	41.7
Bellevue → Ped1	50.75	30.56	45.89
Train → Ped1	51.75	39.72	33.66
Ped2 → Ped2	26.26	44.13	33.26
Ped1 → Ped2	26.25	46.14	38.54
Bellevue → Ped2	26.26	34.63	38.01
Train → Ped2	25.69	41.89	52.51
Bellevue → Bellevue	33.47	51.38	32.24
Ped1 → Bellevue	33.45	45.45	35.25
Ped2 → Bellevue	33.47	40.63	39.12
Train → Bellevue	32.92	49.31	34.35
Train → Train	47.2	42.84	51.47
Ped1 → Train	46.73	46.16	43.96
Ped2 → Train	46.67	49.0	42.68
Bellevue → Train	46.84	51.46	45.89

CDFG Measure on Anomaly Detection evaluation (Example)

Table 2: Partial CDFG Measure (EER: Equal Error Rate) [dos Santos et al., 2019].

Source → Target	Cross-feature	PCA	TCA
Ped2 → Ped1	25.2	5.0	8.44
Belleview → Ped1	17.28	20.82	10.47
Ped1 → Ped2	25.15	10.97	1.14
Belleview → Ped2	7.21	16.75	5.77
Ped1 → Belleview	17.95	10.28	4.43
Ped2 → Belleview	7.21	3.5	5.86

Table 3: Complete CDFG Measure (EER: Equal Error Rate) [dos Santos et al., 2019].

Datasets	Cross-feature	PCA	TCA
(Ped1, Ped2)	25.2	7.99	4.79
(Ped1, Belleview)	17.6	15.6	7.45
(Ped2, Belleview)	7.21	10.12	5.82

CDFG Measure on Anomaly Detection evaluation (Example - Negative Transfer)

Table 4: Partial CDFG Measure (EER: Equal Error Rate) [dos Santos et al., 2019].

Source → Target	Cross-feature	PCA	TCA
Train → Ped1	4.55	3.12	17.81
Train → Ped2	21.51	0.95	1.04
Train → Belleview	14.28	6.47	17.12
Ped1 → Train	4.67	10.99	4.28
Ped2 → Train	20.41	4.87	9.42
Belleview → Train	13.37	0.08	13.65

Table 5: Complete CDFG Measure (EER: Equal Error Rate) [dos Santos et al., 2019].

Datasets	Cross-feature	PCA	TCA
(Train, Ped1)	4.61	7.06	11.0
(Train, Ped2)	21.0	2.91	5.23
(Train, Belleview)	13.8	3.28	15.4

Topics

- Transfer learning concepts
- **Feature extraction using pre-trained networks**
- Fine-tuning network
- Unsupervised and semi-supervised approaches

Convolutional Neural Network

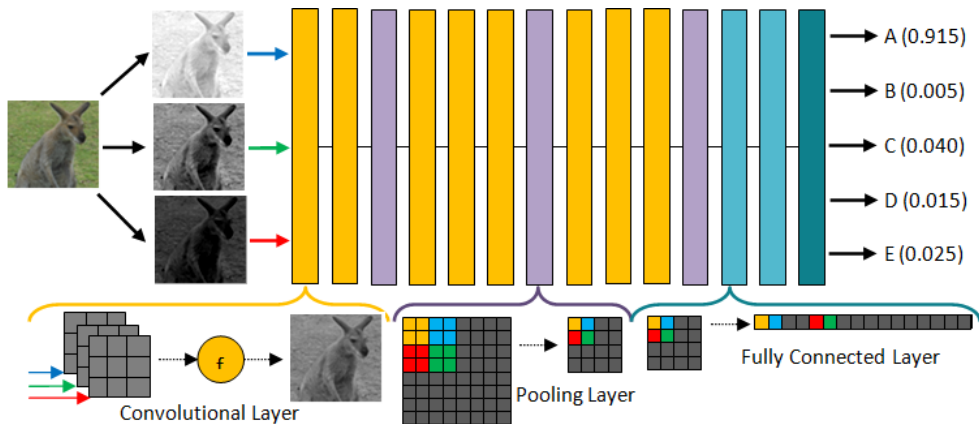


Figure 8: Generic CNN structure.

Convolution Neural Networks - Properties

$$f(x) = f_L(\dots f_2(f_1(x_1, W_1), W_2), \dots W_L) \quad (10)$$

- hierarchical structure;
- each layer provides a distinct feature map;
- low-level (colors and shapes) and high-level features (texture and semantics);
- end-layers are receptive fields of previous layer;
- great flexibility and high levels of cross-domain;

**** Low-level and top-level layers threshold is still uncertain**, with several heuristics prevailing to determine the optimal layer for each problem.

Inception

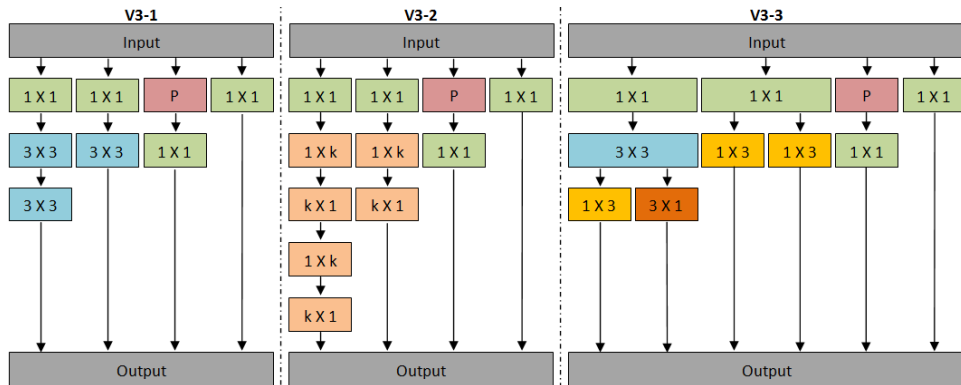


Figure 9: Inception Module. Three different modules were proposed: V3-1 incorporates the idea of small filters being more representative from VGGs; V3-2 applies factorization (orange blocks); and V3-3 increases space dimensionality by bank filters (also in orange blocks). Outputs performed feature map concatenation from each branch. All modules have in common two branches: pooling (P) with 1×1 convolution; and a single 1×1 convolution. Usually, k assumes 7.

ResNet

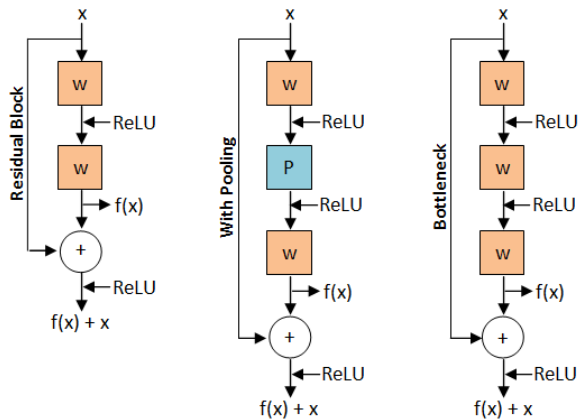


Figure 10: Residual Blocks. All three blocks perform sum of an input x with a data transformation $f(x)$, where w represents a convolutional layer and p defines a pooling layer. Bottleneck proposes to compress the input depth by a reduced number of filters and restore it before the sum.

Feature extraction for an external classifier/detector

- choose the desired pre-trained network;
- choose the desired extraction layer;
- choose the desired classifier/detector.

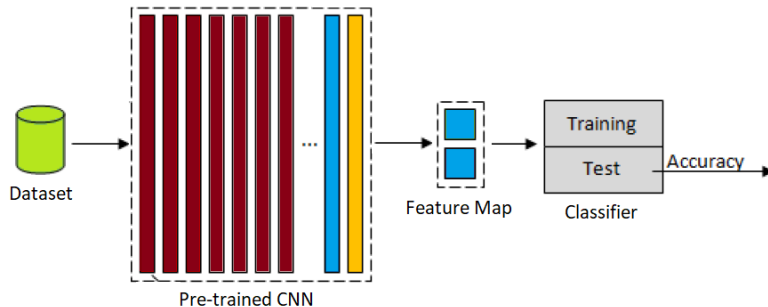


Figure 11: Feature extraction for an external classifier/detector. If necessary, apply cross-validation.
(Code: **FeatureExtraction1**)

Feature extraction and dimensionality reduction for an external classifier/detector

- choose the desired pre-trained network;
- choose the desired extraction layer;
- **choose the dimensionality reduction technique;**
- choose the desired classifier/detector.

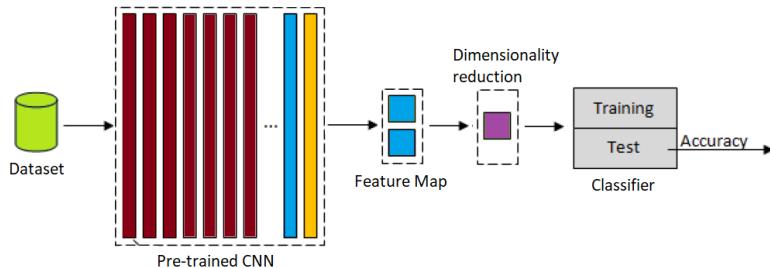


Figure 12: Feature extraction and dimensionality reduction for an external classifier/detector. If necessary, apply cross-validation.

(Code: **FeatureExtraction2**)

Multi-layer feature extraction for an external classifier/detector

- choose the desired pre-trained network(s);
- choose the desired extraction layers;
- **feature map fusion**;
- **< choose the dimensionality reduction technique >**;
- choose the desired classifier/detector.

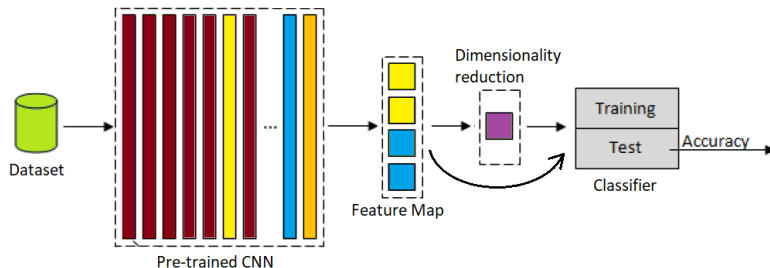


Figure 13: Multi-layer feature extraction for an external classifier/detector. If necessary, apply cross-validation.
(Code: **FeatureExtraction3**)

Considerations: explore more inner layers

Table 6: CNNs pre-trained with ImageNet using PH2 dataset: 20-folds Cross Validation by Balanced Accuracy (%) [dos Santos and Ponti, 2018].

CNN	Layer	Number of Features	Features with Variance	Linear SVM
VGG-19	-2	4096	93.7%	88.5 ± 6.54
	-3	4096	93.7%	88.5 ± 8.53
	-4	25088	86.8%	89.0 ± 6.24
	-5	25088	86.8%	88.5 ± 7.26
	-6	100352	75.2%	91.5 ± 7.92
	-7	100352	92.8%	91.5 ± 6.54
ResNet50	-2	2048	100%	90.0 ± 7.75
	-3	2048	100%	90.5 ± 7.4
	-4	100352	96.3%	91.5 ± 7.92
	-5	100352	100%	91.5 ± 7.26
	-6	100352	100%	90.5 ± 7.4
	-7	100352	100%	90.5 ± 9.73

Considerations: initial layers may enhance performances

Table 7: Classification accuracy (%) of **Supermarket Produce** using feature extraction from ResNet50 pre-trained with ImageNet and **Fruits-360** as source dataset [dos Santos and Ponti, 2019].

Features	Pre-prediction layer	Fusion 1st block	Fusion 2nd block	Fusion 3rd block
256	28.24	36.48	37.18	37.33
192	30.24	36.08	37.33	37.48
128	34.23	38.32	38.87	37.77
96	33.63	39.42	40.52	40.27
64	27.59	37.77	39.62	38.92
Avg.	30.79	37.61	38.7	38.35

Topics

- Transfer learning concepts
- Feature extraction using pre-trained networks
- **Fine-tuning network**
- Unsupervised and semi-supervised approaches

Fine-tuning Network

Fine-tuning

It consists of reusing weights from pre-trained network with large datasets and refining the solution by training the model with the dataset of current task domain.

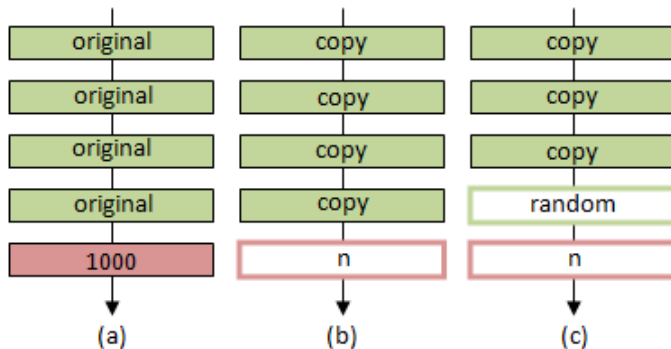


Figure 14: Fine-tuning: modifications in the structure, initialization of parameters, and new training.

Limited resolution

- ResNet50 with top: input of 224 X 224;
- ResNet50 without top: width and height should be no smaller than 197;
- Inception with top: input of 299 X 299;
- Inception without top: width and height should be no smaller than 139;
- a preprocessing step must be applied to reduce or increase resolutions, impacting on loss of information or noisy addition.

Large sample necessary

- sample representativeness;
 - depth of architecture network (fixed);
 - complexity of the task (not easy to measure);
-
- to ensure representativity of the domain is required a large set of instances;
-
- Option 1: Data Augmentation;
 - Option 2: Similar domain.

Overtraining, overfitting, and overparametrization

- defining accurately when to stop the training;
- **overtraining** provides to the network a memorization of data distribution, resulting in poor performance with test samples and avoids generalization to other similar domains (**overfitting**);
- **overparametrization** increases the computational costs;

Network training

Weights Update:

- propagation: weights from the previous training will be influenced by the new training;
- frozen layers: only new layers are updated without changing the previous weights of the network;

Loss Function (measures the progress of network training):

- Cross-entropy will express the penalty for predicting a label \hat{y} in which should be y ;
- Mean Square Error will express the reconstruction error;

Network training (Code: CNNFineTuning)

Optimization Algorithm (adjusts parameters and minimize the loss function):

- Stochastic Gradient Descent (SGD);
- Adaptive Moment Estimation (Adam);

Batch Size (defines the amount of instances loaded in memory):

- some say it should be as large as possible, occupying all memory;
- some say that small batchs allow greater precision in minimizing loss;
- some say that a batch size with 32 examples is ideal, independent of the task;

Number of Iterations (epochs):

- until the network converges measured by the loss function.

Topics

- Transfer learning concepts
- Feature extraction using pre-trained networks
- Fine-tuning network
- **Unsupervised and semi-supervised approaches**

AutoEncoders

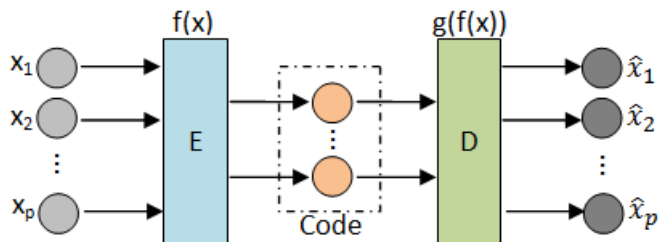


Figure 15: Generic Autoencoder: (E) encoder; (D) decoder.

$$f(x) = \Phi(Wx + b_e) \quad (11)$$

$$g(f(x)) = \Phi(Wf(x) + b_d) \quad (12)$$

Feature extraction from AutoEncoder

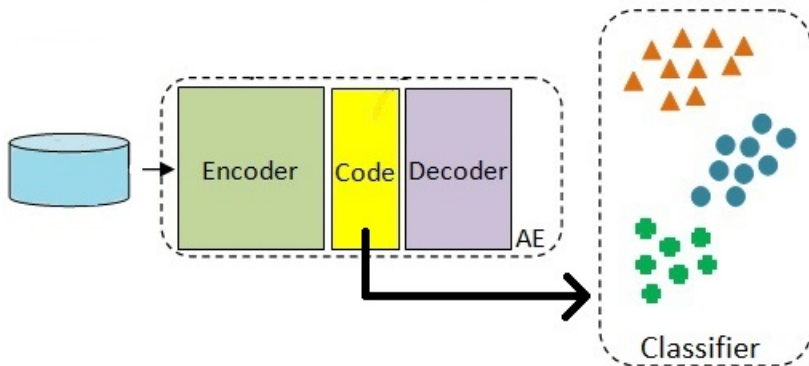


Figure 16: AE feature extraction.
(Code: AutoEncoder)

Feature extraction from AutoEncoder

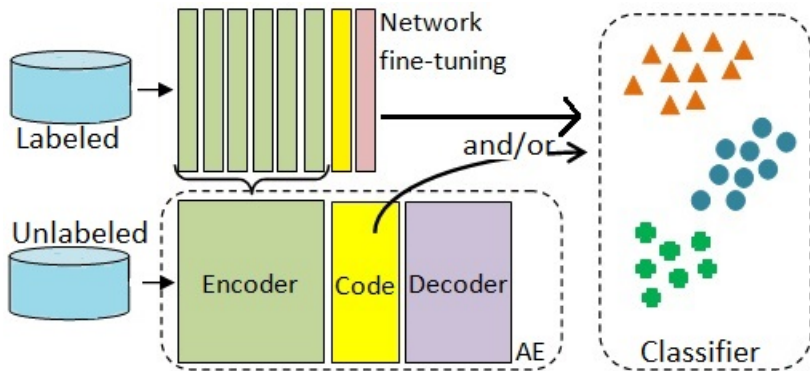


Figure 17: CNN fine-tuning using AE pre-trained encoder.

Final Considerations

- Deep network has been the state-of-the-art in many computer vision applications;
- Transfer learning is a widely exploited resource when looking for model generalization and in situations with low labeled data;
- Feature extraction using pre-trained networks allows generalization, post-processing, and low computational cost;
- Fine-tuning network is a transfer learning technique that provides excellent performance; however, it is necessary to model the network according to the task and the obtained data;
- A necessary concern is to measure the efficiency of the transfer learning, not being restricted to the metrics of each task; CDFG Measure should be used to quantitatively measure generalizable models.






Complementary Readings

- 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, 2016. p. 770–778.
- HOWARD, A. G.; ZHU, M.; CHEN, B.; KALENICHENKO, D.; WANG, W.; WEYAND, T.; ANDREETTO, M.; ADAM, H. **Mobilenets: Efficient convolutional neural networks for mobile vision applications**. arXiv preprint arXiv:1704.04861, 2017.
- PAN, S. J.; YANG, Q. et al. **A survey on transfer learning**. IEEE Transactions on knowledge and data engineering, Institute of Electrical and Electronics Engineers, Inc., 345 E. 47 th St. NY NY 10017-2394 USA, v. 22, n. 10, p. 1345–1359, 2010.
- PONTI, M.; RIBEIRO, L. S.; NAZARE, T. S.; BUI, T.; COLLOMOSSE, J. **Everything you wanted to know about deep learning for computer vision but were afraid to ask**. In: 30th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T 2017), 2017. p. 17–41.
- YOSINSKI, J.; CLUNE, J.; BENGIO, Y.; LIPSON, H. **How transferable are features in deep neural networks?** In: Advances in neural information processing systems, 2014. p. 3320–3328.

Libraries documentation

- **Keras:** <https://keras.io/api/>
- **Sklearn:** <https://scikit-learn.org/stable/>
- **Matplotlib:** <https://matplotlib.org/>
- **Colab:** <https://colab.research.google.com/>

Referencies I

-  dos Santos, F. P. and Ponti, M. A. (2018).
Robust feature spaces from pre-trained deep network layers for skin lesion classification.
In 2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), pages 189–196. IEEE.
-  dos Santos, F. P. and Ponti, M. A. (2019).
Alignment of local and global features from multiple layers of convolutional neural network for image classification.
In 2019 32st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI). IEEE.
-  dos Santos, F. P., Ribeiro, L. S., and Ponti, M. A. (2019).
Generalization of feature embeddings transferred from different video anomaly detection domains.
Journal of Visual Communication and Image Representation, 60:407–416.
-  Goodfellow, I. J., Bengio, Y., and Courville, A. (2016).
Deep learning.
MIT Press.
-  Torrey, L. and Shavlik, J. (2010).
Transfer learning.
In Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques, pages 242–264. IGI Global.

Aprendendo representações com Deep Learning: arquiteturas, treinamento e transferência de aprendizado

Fernando Pereira dos Santos

fernando_persan@alumni.usp.br
fernando.santos@birdie.ai
<https://www.linkedin.com/in/fernando-persan/>

October, 2020

