BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

COMPUTER VISION WITH ACTIVE LEARNING

POČÍTAČOVÉ VIDĚNÍ S AKTIVNÍM UČENÍM

PHD THESIS
DISERTAČNÍ PRÁCE

AUTHOR **MARTIN KOLÁŘ, Ph.D.**
AUTOR PRÁCE

SUPERVISOR **prof. Dr. Ing. PAVEL ZEMČÍK**
ŠKOLITEL

BRNO 2020

Abstract

Machine Vision methods benefit from improving models, tuning trained parameters, or labeling representative data. In a series of experiments, this work validates the hypothesis that Active Learning improves the accuracy of these models. By extending the pseudolabel framework to Active Learning, this work includes a One-shot-learning approach to learn novel image categories by utilising an algorithmic recommender, an online Graphical User Interface to optimise the online Exploration/Exploitation tradeoff for tagging, and a two-step offline binary Active Learning framework to improve the quality of data used for Font Capture. By demonstrating the benefit of Active Learning in these approaches, this work contributes to the hypothesis, as well as concrete Machine Vision applications.

Abstrakt

Metody strojového vidění se zdokonalují zlepšením modelů, laděním trénovaných parametrů nebo anotací reprezentativních dat. Tato práce řadou experimentů potvrzuje hypotézu že aktivní učení zvyšuje přesnost těchto modelů. Rozšířením přistupu pseudolabelů o aktivní učení přispívá tato práce přístupem „one-shot-learning“ k učení nových kategorií obrazů s použitím algoritmických doporučení, dále online grafickým uživatelským rozhraním pro optimalizaci dilema Exploration/Exploitation pro online tagování, a dvoukrokovým offline binárním přístupem aktivního učení pro zlepšení kvality dat používaných pro snímání fontů. Tím, že demonstruje přínos aktivního učení v těchto přístupech, přispívá tato práce k hypotéze i konkrétním aplikacím strojového vidění.

Keywords

Computer Vision, Object Classification, Semi-supervised Learning, Active Learning, Transfer Learning

Klíčová slova

Počítačové vidění, Detekce objektů, Částečné učení s učitelem, aktivní učení, přenášené učení

Reference

KOLÁŘ, Martin. *Computer Vision with Active Learning*. Brno, 2020. PhD thesis. Brno University of Technology, Faculty of Information Technology. Supervisor prof. Dr. Ing. Pavel Zemčík

Computer Vision with Active Learning

Declaration

I declare that this dissertation thesis is my original work and that I have written it under the supervision of prof. Dr. Ing. Pavel Zemčík. All sources and literature that I have used during elaboration of the thesis are correctly cited with complete reference to the corresponding sources.

.....

Martin Kolář

September 14, 2020

Acknowledgements

I would like to thank my friends, colleagues, family, and supervisor for their unending support.

Contents

CHAPTER 1

Introduction

Machine Learning in general and Computer Vision in particular are highly sensitive to the amount and quality of data presented. Reproducible improvements in results and abilities of created models can be wholly attributed to the following factors: a model's prior and ability to fit, a model's actual fit to data, and representativity of training data.

With the vast majority of research focused on the first factor, and the vast majority of engineering work focusing on the second, the third factor receives far less attention. As quality data for various critical tasks is known to be costly to create, this work focuses on how this can be done most effectively. The optimisation of expert labelling and assessment of the resulting benefits is achieved via Active Learning.

A hypothesis regarding the benefit of Active Learning in Machine Vision is experimentally validated under these scenarios: when the labelling oracle is a human expert, and when the oracle is a pre-trained algorithm for another task. A large dataset, such as personal photos or hundreds of thousands of fonts, is most effectively labelled by using a human-in-the-loop approach. Facilitated by per-sample certainty analysis during training, the expert labelling effort achieves higher label quality with significantly fewer annotations. The second case, where existing models can preprocess useful information, requires systematic labelling and confidence tracking, or practical mappings. I have successfully used these approaches in my own published work, as presented in this thesis.

The hypothesis is experimentally validated in several contemporary scenarios, such as fully automated labelling (section 5.2) to create a dataset with combined information not available elsewhere, but with models trained on several other datasets. My work demonstrates that filtering data by retraining a model to select valuable examples only is shown to be usable to minimise human effort in creating new, useful datasets, and models.

In addition to four cases validating the hypothesis that Active Learning benefits contemporary Computer Vision, at the core of this work lies another practical contribution: an im-

provement to the accuracy to work ratio achieved through a new pseudolabels Active Learning framework to integrate labelling by existing models, human experts, and a trained agent.

The thesis is structured as follows: this introduction is followed by an overview of Active Learning, with relevant prior work, including my own, in chapter 2. Next, chapter 3 presents an overview of relevant work in Machine Vision, including my own. Chapter 4 discusses the hypothesis to be validated, and the approach taken. The work done in Active Learning for Machine Vision is them presented in four sections of chapter 5, which is the core of the thesis. Finally, chapter 6 gives an overview of applications and future work, followed by the conclusion.

CHAPTER 2

Related Work in Active Learning

Active Learning is the process of selecting which data needs to get an expert label, either by a human or by another algorithm. However, this can correspond to a variety of scenarios, and be handled in a panoply of ways [KSF17, BRK19]. Due to the breadth of the subject, only approaches relevant to the contents of this work are dealt with here. This chapter presents an overview of techniques, gives some definitions, presents advances in the field, and finally describes shortcomings and how these can be addressed.

2.1 Overview

The vast majority of Machine Learning applications relies on Supervised Learning, which is the fitting of a model to extrapolate given labelled data. This section uses standard notation used in Active Learning literature [Mun20]. Formally, a model M is trained to approximate y given x as follows

$$y_p = M \times x \tag{2.1}$$

$$\underset{M}{\operatorname{argmin}}\, L(y_p, y_d) \tag{2.2}$$

where y_d is the dataset annotation, and y_p is the model M prediction given x.

In other applications, Unsupervised Learning is used. Under the unsupervised framework, unlabelled data is used to create a model. As before, M is trained to minimise a loss function L as follows

$$\underset{M}{\operatorname{argmin}}\, L(M \times x) \tag{2.3}$$

A typical Supervised Learning task is the classification of an image by the class of object it contains, and an unsupervised task can be the prediction of the next character in a text sequence [LDXT11, LDTX12]. Both approaches create a model which can be applied to new unseen data. Sometimes, the labelled data can be supplemented by a set of unlabelled data, and this combination is used under the Semi-supervised Learning framework. This can be formulated as follows

$$\underset{M}{\operatorname{argmin}} \left(L_u(M_u \times x_u) + L_s(y_p, y_d)\right) \mid y_p = M_s \times x_l \tag{2.4}$$

where M_u is the portion of the model trained on unlabelled data, and M_s is the portion of the model trained on labelled data

$$M = M_u \cup M_s \tag{2.5}$$

By extending the semi-supervised approach to select data to be labelled during training, the approach becomes Active Learning with an expert E on selected unlabelled data x_s:

$$y_e = E(x_s) \tag{2.6}$$

$$\underset{M}{\operatorname{argmin}} \left(L_u(M_u \times x_u), L_s(y_e = M_s \times x_s), L_a(x_s = S \times x)\right) \tag{2.7}$$

$$M = M_u \cup M_s \cup S \tag{2.8}$$

Where S is the Active Learning component of the model which performs the selection. This process is iterative, selecting data as it is labelled:

$$\begin{aligned} \forall i \quad \mid \quad X_{c,i} &= S_i(M, \rho_i, X_{s,i}, X_{u,i}) \\ X_{s,i+1} &= X_{s,i} \cup E(X_{c,i}) \end{aligned} \tag{2.9}$$

Furthermore, the training process may have various labeling options, in the form of multiple choices of experts. This creates an exploration-exploitation trade-off dilemma, becoming a Reinforcement Learning problem, modelled generally as a Markov decision process:

$$\begin{aligned} R_a(s, s') \\ P_a(s, s') = Pr(s_{t+1} = s' | s_t = s, a_t = a) \end{aligned} \tag{2.10}$$

Where R_a is the immediate reward after transition from state s to state s' via action a. In the context of Active Learning, this can be simplified by setting the reward equal to the new total model accuracy, and dependent on the selected expert E_n and datapoints x_s

$$R_a(x_s) = L_u(M_u \times x_u), L_s(y_e = M_s \times x_s), L_a(x_s = M_a \times x) \tag{2.11}$$

Depending on the assumptions made about the data, this could be presented as Weakly Supervised Multiple Instance Learning (WSMIL), a subproblem of Semi-supervised Learning.

By making the assumption that at least one of the queried images belongs to a given class, training with unlabelled data becomes WSMIL [VG08]. This approach has been coupled with the traditional image classification approach of a dividing hyperplane in a feature histogram hyperspace.

Investing time and resources into labelling a portion of a dataset for use in a Semi-supervised Learning framework leads to inefficiencies, as some data points are more informative than others. As a trivial example, take the labelling 100 samples of an unlabelled dataset of 1000 datapoints, where 100 datapoints are in class A and 900 in class B. If performed at random, probably only ten datapoints of class A will be labelled. Active Learning would enable a better distribution of expert labelling effort, so that a more representative sample of A is labelled.

Thus, Active Learning enables the creation of more accurate machine learning models. The basic principles of Active Learning are applicable to advanced scenarios, such as extrapolating single-task learning, where a model learns a single task, to Multitask Learning, where multiple models are joined to share training parameters over several tasks.

- to make Machine Learning more accurate
- to make humans more efficient
- achieving desired Machine Learning accuracy faster
- making humans more accurate

This last point is derived from another application of Active Learning, one with an orthogonal and complementary use-case to optimal datapoint selection. Active Learning can also prepare the presented data in such a way as to make human expert input more effective, for instance by highlighting relevant sections of a video to be labelled, or by presenting extracted features instead of raw data.

The Active Learning approach lends itself well to combination with Transfer Learning, either by fine-tuning an existing model or by applying an existing model to act as an expert annotator oracle.

Finally, Active Learning is also used to address the complex issue of human errors in training data. When Supervised Learning models reach the same accuracy as the humans who labelled the data, for example a 1% error rate, breaking this barrier becomes challenging given this same data. When predictions on training data are predicted as inaccurate, an Active Learning system may trigger a closer examination or examination by multiple experts. By utilising information such as labeller identity and error rate, a system can more accurately sift out human error which would otherwise propagate into the model, and into test-time predictions.

2.2 Uncertainty Sampling

Also called exploitation, uncertainty sampling aims to seek labels for datapoints near the decision boundary of a trained model.

Uncertainty sampling can be performed in several ways. Some of these are least confidence sampling, margin of confidence sampling, ratio of confidence sampling, and entropy-based sampling. All these require a probability or confidence, which is weighted in different ways. However, only Bayesian and probabilistic models are capable of producing a confidence or confidence interval for every prediction made. Therefore, Bayesian models are ideally suited for Active Learning in the context of Uncertainty sampling, as well as methods presented below.

However, Bayesian models achieve lower accuracy on critical tasks compared to Deep Learning approaches. Especially deep neural networks, which inherently defy interpretability, and for these the confidence is substituted with a per-label score. Neural networks for classification can be trained to output a value in the $0-1$ interval, and these values can be normalised across classes to satisfy the criteria of a discrete probability distribution:

$$\begin{aligned} \forall x \in X, \quad & 0 \leq P(x) \leq 1 \\ & \sum_{x \in X} P(x) = 1 \end{aligned} \tag{2.12}$$

For example, Active Thompson Sampling (ATS) [BLU+14] assigns a distribution to the pool, and samples one point from this distribution for labelling.

2.3 Diversity Sampling

Also called exploration, „Representative Sampling“, „Outlier Detection“, or „Anomaly Detection“, the diversity sampling approach requests labels for datapoints which are unknown or uncertain for the model being trained. Rather than merely dividing data into labelled and unlabelled, diversity sampling quantifies whether unlabelled datapoints are probably known or probably unknown. The ‘known unknown’ and ‘unknown unknown’ terminology has been adapted from Chemistry [LWPT11] to Machine Learning to denote unlabelled sections of the dataset which are likely classified correctly by the existing model, versus those which are likely not classified correctly. Where uncertainty sampling deals with quantifying what a model knows about the unknown, diversity sampling is a complementary approach which quantifies what the model does not know about the unknown [Mun20].

The typical diversity sampling approaches utilise Model-based Outlier Sampling, Cluster-based Sampling, Representative Sampling, and Sampling for Real-World Diversity. These model the data, the machine learning model being trained, or enforce the creation of an objectively more varied dataset.

Current approaches maintain a committee of models with different hypotheses, whose votes are aggregated to produce most likely anomalies [DWD+16, Dzz15]. This *Query-by-Committee* [SCF08] approach is model agnostic, while simple enough to be incorporated for a wide variety of models, as long as these maintain an outlier rating of unlabelled training data points.

Another widely used approach is to fit a Support Vector Machine (SVM) to labelled data, and used the margin, W, of each unlabelled datum in $T_{U,i}$ and treat W as an n-dimensional measure

of certainty. This measure can be converted to a single scalar by computing the L distance, such as $L-1$ (Manhattan distance), $L-2$ (euclidean distance), or $L-\infty$ (supremum norm).

2.3.1 Test-time Anomaly Detection

Diversity sampling is concerned with the training-time evaluation of data, which may be referred to as training-time anomaly detection. For test-time, this task is closely linked to Refinement, where external annotators select and label samples to retrain a model with [LB97], but where the model is expected to perform this function independently [GSC00], or in concert with annotators [ONJ13].

Measuring the certainty of a label can be performed just as it was done at training time, by model-based outlier sampling, cluster-based sampling, representative sampling, or sampling for Real-World Diversity.

Model-based outlier computation can be performed on models which explicitly compute this information, such as Bayesian models or non-parametric models. When the model produces scores in an interval, as with SVMs [CV95], neural networks, or random forests [Ho95], these may not reflect confidence. While the score is generally interpreted as such, there is no guarantee as to the relationship between datapoints of equal certainty or class probability, unlike fully-connected statistical Bayesian models.

Cluster-based sampling may be more appropriate for these latter cases. By agglomerating data into clusters, and analysing unlabeled points together, noisy statistical estimates can be smoothed and sampled more accurately. A multi-clustering approach can help smooth the distributions further, as in the following case, where a multitude of models helps to find anomalies.

In representative sampling, multiple models are taken and compared. Unlike cluster-based sampling, the joining does not take place over data, but over models. In order to minimise the amount of work and the overall complexity, the same model can be trained with various input randomisations, with leave-one-out correlation training, or on sequentially shuffled data to produce different but equivalent training parameters on the data. These models can then be uniformly joined to produce smoother labelling.

Finally, Real-World Diversity is another family of approaches to identify and reduce bias for arbitrary Machine Learning models. In essence, these aim at creating a prior over models and a set of data such that they are increasingly representative. Performing this requires expert knowledge of the demographics of the data, as well as an ideal application of models.

2.3.2 Federated Learning

Gathering data with Active Learning may be some disadvantages, especially where the collection and processing of data are tightly interwoven [KMY+16]. For certain tasks, the centralised collection of training data may be impractical, undesirable, or made impossible by their private nature. For tasks such as onscreen mobile keyboard predictions [MMR+16], Federated Learning can offer significant advantages, resulting in higher accuracy.

Federated Learning is a form of online learning, where all data is not available centrally at preparation time [BEG+19]. For example, a language model may begin by being initialised on idealised data, such as Wikipedia text. Next, the model is distributed to devices, because the application requires the prediction of real text as users type it. Next, each user gathers their own non-iid data by typing on their keyboard and positively or negatively labelling the suggestions as they are or are not selected [ZLL+18].

When the device is charging, a procedure retrains the recurrent neural network with the new data. This fine-tuning is not performed centrally, so each user has a differently updated model. In order to bring these updates together without jeopardising privacy, the updates are uploaded and centrally weighted, with the resulting average taken as updates to a centralised model.

With this general Federated Learning approach, it is possible to update image annotators, face detectors, or any other application that may otherwise require the gathering of a static dataset, which would then be centrally updated via traditional Active Learning. Another advantage of this approach is that an essential portion of the processing is performed at the client-side, reducing the server load for improving the model. In an era of powerful portable computers, this Active Learning approach is increasingly attractive for all scenarios where users are incentivised to annotate data.

2.4 Restricted Boltzmann Machine

A typical model for multi-class labelling with confidences is the Restricted Boltzmann Machine [Smo86]. RBM is an undirected bipartite graphical model [CPH05]. It defines a probability distribution over a vector of visible variables $\mathbf{v}$ and a vector of hidden variables $\mathbf{h}$. In the RBM model, the visible variables are independent of each other when the hidden variables are observed and vice versa.

For modelling dependencies among semantic tags, the visible variables $\mathbf{v}$, each corresponding to the presence of a tag, are binary. In this work, the hidden variables $\mathbf{h}$ are binary as well.

The joint probability over $\mathbf{v}$ and $\mathbf{h}$ is defined as

$$p(\mathbf{v},\mathbf{h}) = \frac{exp(-E(\mathbf{v},\mathbf{h}))}{Z}, \tag{2.13}$$

where Z is a normalisation constant and E is energy function given by

$$E(\mathbf{v},\mathbf{h}) = -\mathbf{v}^\top \mathbf{W}\mathbf{h} - \mathbf{v}^\top \mathbf{b}^v - \mathbf{h}^\top \mathbf{b}^h, \tag{2.14}$$

where (W) is a matrix of weights between elements of $\mathbf{v}$ and $\mathbf{h}$, and $\mathbf{b}^v$ and $\mathbf{b}^h$ are biases of visible respective hidden variables. Conditional dependencies between the visible and hidden variables are expressed as

$$p(\mathbf{h}|\mathbf{v}) = \sigma\left(\mathbf{W}\mathbf{v} - \mathbf{b}^v\right) \text{ and } p(\mathbf{v}|\mathbf{h}) = \sigma\left(\mathbf{W}^\top \mathbf{h} - \mathbf{b}^h\right), \tag{2.15}$$

where $\sigma()$ is a sigmoid function.

As a generative model, RBM could be trained using maximum likelihood. However, derivatives of the likelihood are intractable. To overcome this problem, Hinton [Hin02] introduced a practical approximation called *Contrastive Divergence* (CD). The CD algorithm computes gradients for optimisation as

$$\nabla \mathbf{W} = \langle \mathbf{vh} \rangle_{data} - \langle \mathbf{vh} \rangle_{recon} \tag{2.16}$$

$$\nabla \mathbf{b}^{v} = \langle \mathbf{v} \rangle_{data} - \langle \mathbf{v} \rangle_{recon} \tag{2.17}$$

$$\nabla \mathbf{b}^{h} = \langle \mathbf{h} \rangle_{data} - \langle \mathbf{h} \rangle_{recon}, \tag{2.18}$$

where $\langle . \rangle_{data}$ are expectations with respect to the distribution of data and $\langle . \rangle_{recon}$ are expectations with respect to the distribution of reconstructed data. The reconstructed data is obtained by starting with a data vector on visible variables, and sampling first from distribution $p(\mathbf{h}|\mathbf{v})$ and then $p(\mathbf{v}|\mathbf{h})$ (Equation 2.15).

In the context of tag suggestion, the task of RBM is to provide marginal probabilities of unobserved tags which constitute the visible variables $\mathbf{v}$ as more and more tags become observed (by actions of a user). Several algorithms could solve inference in the RBM model. *Gibbs sampling* was chosen. It draws several samples from the RBM distribution. The means of marginal distributions $E(p(v_i))$ can then be computed from the samples. Gibbs sampling starts by assigning random values to unobserved variables, and a sample is obtained by iterating between computing $p(\mathbf{h}|\mathbf{v})$ (Equation 2.15) and sampling from it, followed by computing $p(\mathbf{v}|\mathbf{h})$.

As it is not practical or desirable to obtain a large training dataset where presence or absence of all tags for all images would be known due to a large number of possible tags (hundreds or thousands), inevitably, such dataset has to have sparse annotations. Furthermore, the learning algorithm has to handle situations where a potentially large portion of the tags is unobserved. Several methods for handling missing training data in the context of RBM were proposed. Single missing value can be easily filled by sampling from its exact conditional distribution (it is known for single unobserved variable). More missing values can be treated in the same way as the other parameters [HOT06] if they are updated often during learning. This approach is efficient only on training sets of limited size. Salakhundinov et al. [SMH07] introduced a radical way of dealing with missing values by using RBM's with different numbers of visible units for different training cases. This approach is able to handle very sparse data; however, it no longer produces a single RBM model.

In this work, Gibbs sampling was used to fill the unobserved values in the training data. For the CD gradients (Equation 2.16), the data means $\langle . \rangle_{data}$ have to be computed. This can be done by drawing samples from the distribution of the unobserved visible variables conditioned on the observed visible variables. This distribution is not known during learning of the RBM model. However, the current imperfect RBM model can be used instead as an approximation. When a sample from the distribution of the visible data is obtained, the CD algorithm proceeds exactly as described in Section 2.4.

2.5 Training with Simulated Data

It is known that Active Learning can benefit from simulated data [KSF17], and this result is used to create improved embeddings for electron microscopy segmentation.

A Scanning Electron Microscope (SEM) equipped with an Energy Dispersive X-ray spectroscopy (EDS) detector is a well-established tool to analyse the chemical composition of samples with a spatial resolution of the order of µm [GNM+18]. It is widely used in material science and geological/mineralogical research. Naturally, proper recognition of phase boundaries is essential for meaningful and trustworthy analytical results and interpretation. The phases are generally recognised 1) based on their BSE brightness value only, or 2) using a combination of BSE and EDS data. On the other hand, acquisition of EDS data is orders of magnitude slower than the acquisition of BSE data (milliseconds per pixel compared to microseconds per pixel, respectively), thus inordinately prolonging the analysis. To keep the analysis reasonably short one could either acquire the EDS data in a coarser grid then the BSE [HGS+18, MF13], or acquire only a relatively small number of X-ray events from each pixel (e.g. 1,000) or combine both. The disadvantage is that the low-count EDS spectra are prone to statistical noise, which is typically higher than spectral differences of the phases of interest. Classifying such data pixel-by-pixel inevitably leads to artefacts such as misclassified or unclassified pixels.

A Graph-based Deep Learning Segmentation method (GDLS) is developed, such that is has the capacity to correctly distinguish chemically similar phases in a BSE image, combined with noisy and sparsely occurring EDS data, while still maintaining a sizeable average segment size. This method is, therefore, of great benefit with regard to the robustness of results and speed of analysis.

The proposed segmentation method takes one input field composed of a densely sampled BSE array and sparsely sampled EDS data with known measurement locations. Each EDS measurement is a spectrum with 3,000 bins corresponding to energies from 0 to 30,000 eV with 10 eV steps. The values in the BSE array are calibrated, with a maximal value of 2^{16}. Values in the EDS spectra are positive integers corresponding to the number of detected events at each energy. An example field is seen in Figure 2.1.

The output of the segmentation process is a dense array with pixel-wise labels (integer values) where pixels with the same value belong to the same segment. The core of the proposed method is to determine if spatially close EDS measurements capture the same material, in which case they belong to the same segment and should be labelled with a common label. A graph with EDS measurements represented as vertices is constructed. Edges connect spatially close measurements in a planar graph obtained by Voronoi analysis [Bow81]. A deep learning model assigns a value to each edge. This value corresponds to differences in BSE and distances between hyperspace representations of EDS spectra. Edges with high differences are removed, and the resulting graph components constitute separate segments — labels. Final dense labels are obtained by a dense Markov Random Field operating over the BSE array with labels initialised from the graph. An example of a graph and segmentation is in Figure 2.2. The individual steps of the algorithm are described next. They are projection of spectra to latent space, graph

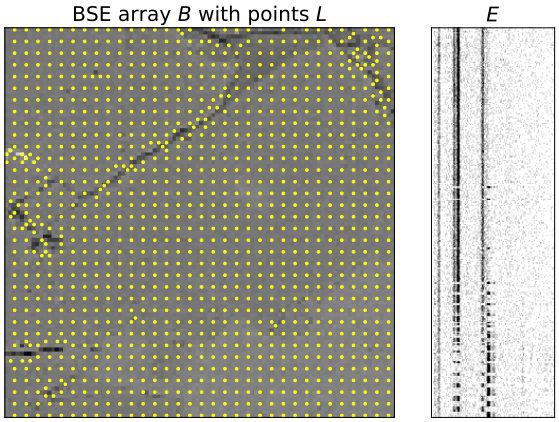


Figure 2.1: The field is a hyperspectral image with sparsely sampled data. The locations L of EDS measurements are superimposed on the BSE array B. The matrix E shows the actual spectra in rows. Details in section 2.5.1

.

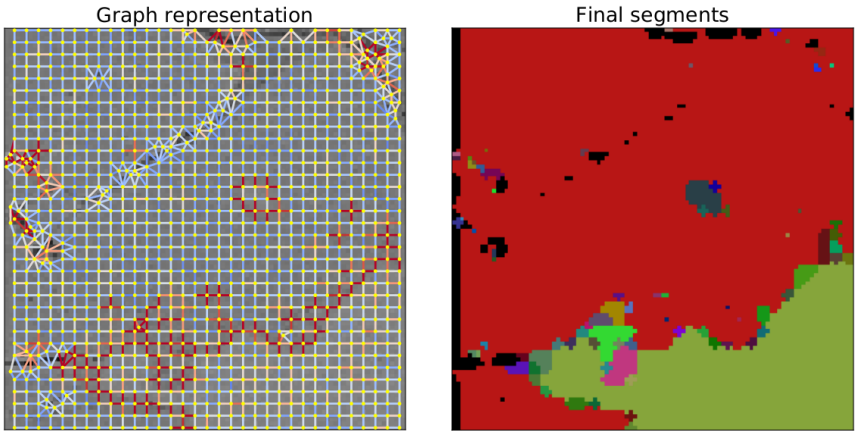


Figure 2.2: The proposed method constructs a graph representation of the field (edge color encodes δ_z, see Section 2.5.6) and decomposes the graph to obtain compact segments. Details in the text.

construction and calculation of edge values, graph component analysis, and Markov Random Field segmentation.

2.5.1 Field Representation

A field F (see Figure 2.1, left) is represented as a triplet $F = (B, E, L)$. B is $H \times W$ array with BSE measurements (an image of width W and height H). E and L are matrices containing N rows with EDS measurements

$$E = \begin{pmatrix} s_1 \\ \vdots \\ s_N \end{pmatrix} \quad L = \begin{pmatrix} l_1 \\ \vdots \\ l_N \end{pmatrix}$$

where E is $N \times D$ matrix with N spectra with dimensionality D, and L is a $N \times 2$ matrix with locations of EDS measurements in the BSE array. The spectrum s_i corresponds to location l_i in the BSE array.

The proposed method assumes full coverage by B, and sparse coverage by EDS measurements, whose locations are determined at scan-time by a separate mechanism internal to the electron microscope. Although the proposed method poses no restrictions on the locations of the sparse EDS measurements, they are best placed at regular intervals on discrete materials, and ideally do not miss small separate grains.

2.5.2 Mineral Dataset

For training the embedding model, a dataset is used which contains 4,507 distinct materials relevant for the application. Each material is characterised by its ideal spectrum. During the training, spectra are generated from the ideal profiles by random sampling with a random number of events to simulate data from a microscope. These spectra are close to what is captured by the microscope, but it is not perfect since the ideal profiles do not include recurring artefacts of pulse processing electronics like sum peaks or escape peaks, and other unmodeled effects [Rit09]. These differences influence the overall performance of the proposed method on the real data especially in situations where multiple materials are mixed (e.g. on boundaries between two materials) since are trained on pure spectra with no “deformations”.

2.5.3 Spectral Decomposition with Deep Learning

The key part of the algorithm is to determine if two spectra capture the same material when they are composed of the same elements in the same ratio [GK72]. While this can be solved with a relatively low error by, e.g., mean absolute difference or a two-sample K-S test on high count spectral data [VSK+19], in the presence of noise due to a low number of X-ray events per pixel the task becomes difficult. In this solution, the spectra are processed with a deep learning model to provide a compact descriptor which can differentiate between spectra with different compositions [WWM+06]. The descriptor is obtained by embedding to a low dimensional space – latent space $\mathcal{Z}$ as shown in Figure 2.3.

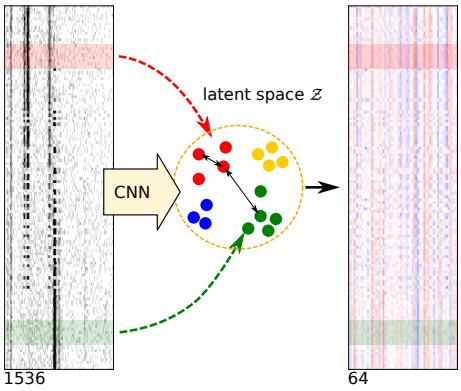


Figure 2.3: The original spectra are projected by CNN to a latent space $\mathcal{Z}$ with 64 dimensions, where similar materials are mapped to similar locations

Transforming EDS measurements into this latent space serves as a preprocessing step to separate image-space segmentation from material composition and characteristic line differentiation, but this model is made on resampled ideal spectra, which cannot contain all noise and variations of real measurements. By training the mapping to differentiate thousands of randomly selected materials, the network learns to generalise even unseen materials. Therefore, during segmentation, which is not trained for specific materials, the presented approach is not limited to previously seen minerals and solid solutions. This is demonstrated by visual analysis of chemically-distinct domains with different Nb/Ta ratio in grains of a columbite-tantalite solid solution are properly segmented despite the fact that no spectrum corresponding to such chemical composition was in the training dataset.

2.5.4 Neural Network Design

The convolutional neural network (CNN, see Figure 2.4) comprises three parts — *normalization*, *feature extraction* and *embedding*. The input of the network is the vector of the spectrum s with dimensionality D followed by normalisation with mean μ and standard deviation σ calculated from ground truth spectra. The 200 eV to 15,560 eV spectral band with 10 eV bins is used. Therefore, the input to the network are spectra with $D = 1,536$. There are two reasons for this: first, coefficients outside this band are mostly uninformative and second, reducing the number of coefficients reduces the number of network parameters and this positively affects the execution speed of the network. However, nothing prevents using the whole spectrum. Feature extraction contains 3 blocks with a sequence of convolution, batch normalization and ReLU activation [NH10] followed by average pooling. Embedding processes the features with a sequence of dense layers and normalises the output to unit length, producing a latent vector z.

Although almost any network design can be used here, a simple network inspired by VGG architecture [SZ14] is chosen, mainly for performance reasons. On a single field, usually, tens of thousands of measurements must be processed (in extreme cases it can be hundreds of thousands). This network can produce its output in a matter of seconds on low-end hardware which is still acceptable.

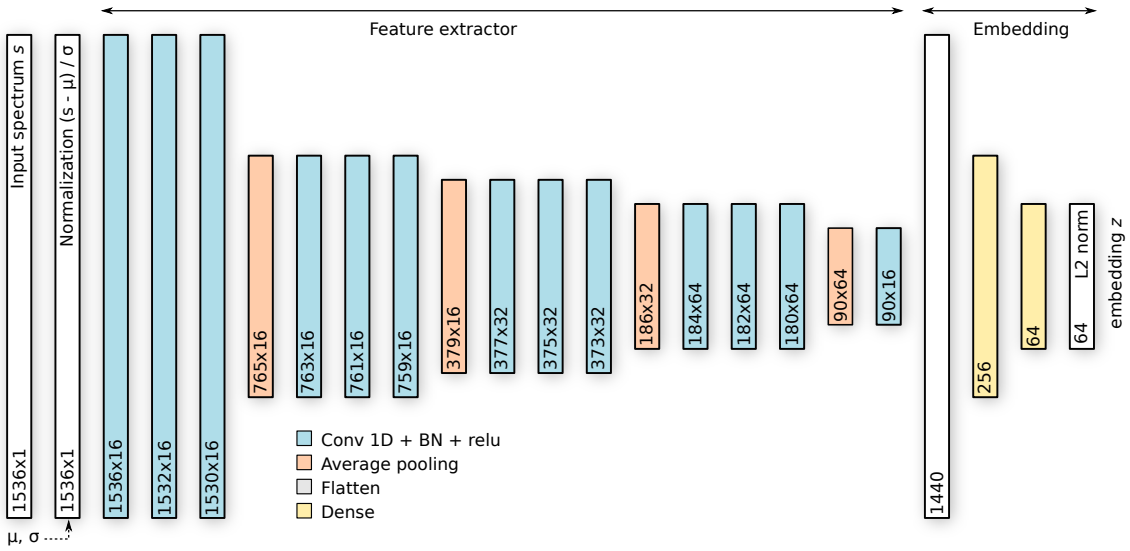


Figure 2.4: The neural network for mapping spectral information to latent space $\mathcal{Z}$. It processes the input s with normalization, feature extraction and embedding and produces the vector z.

2.5.5 Embedding to Low-Dimensional Space

We train a neural network model M which produces a unit vector $z = M(s)$ for a spectrum s. The output dimensionality is a free parameter (64 in these experiments). The spectral information is used as an input and material class (i.e. its integer ID) as an output and optimize the network with ADAM [KB15] optimizer with Semi-hard Triplet loss [SKP15].

The main property of the latent space $\mathcal{Z}$ is that the difference $||M(s_1) - M(s_2)||_2$ is close to zero if s_1 and s_2 are from the same material and it is larger if the materials are different, whatever the materials are. By multiplying matrices M and E, individual EDS measurements s are transformed into the spectrum decomposition z.

2.5.6 Graph Construction and Labelling

From the known locations of EDS measurements in the field, a planar graph $G = (V, E)$ is constructed by Voronoi analysis [Bow81]. Vertices $V = (l_i, b_i, z_i)$ are locations of the measurements, each assigned with its location l_i, BSE value b_i and spectrum decomposition z_i obtained by the neural network. Edges are formed between spatially close measurements. The edges between measurements with distance larger than the $2\times$ sampling distance are not considered. Each edge $E_k = (i, j)$ is assigned with values δ^b_k and δ^z_k capturing difference in BSE and latent space representation of the spectrum [JVK+20].

$$\delta^b_k = |b_i - b_j|/2^{16} \tag{2.19}$$

$$\delta^z_k = ||z_i - z_j||_2 \tag{2.20}$$

Where the value of δ^b is in the $(0, 1)$ interval and $\delta^z >= 0$. Edges with $\delta^b_k > \tau_b$ or $\delta^z_k > \tau_z$ are removed, which disconnects measurements with high difference in BSE or spectrum. The

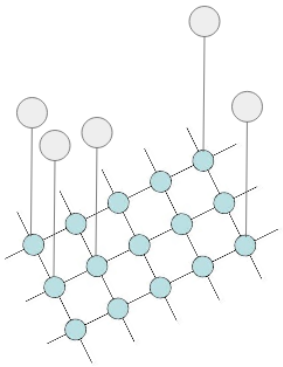


Figure 2.5: Dense per-pixel measurements (blue), sparse markers (gray), and relationships in a Markov Random Field

parameters τ_d and τ_z can be tuned by the user to obtain different segmentations of the input data. The components of the resulting graph then constitute sets of measurements with the same material. As a result, each vertex is assigned a value $c_i \in N$ which is the label of the component it belongs to. These labels are the starting point for the Markov Field, which generates the final pixel-wise label map.

2.5.7 Pixel Labelling with Dense MRF

Markov Random Field segmentation [Li94] is a flexible optimisation method to create unlabelled segments given neighbouring pixel similarities and a sparse marker initialisation. The field of pixel-wise BSE measurements is set on a fully connected grid, and the segmentation map is initialised with known labels from EDS graph analysis. By using *a priori* probability of sampling different measurements from the same distribution (class), and the probability of adjacency of different classes, the entire system becomes a joint distribution whose *a posteriori* distribution can be maximised by computing conjugate gradients for a given segmentation estimate. The distribution is iteratively optimised until convergence, where classes correspond to individual class probabilities. The maximum class at each pixel is taken to be its class, producing full segmentation.

2.6 Superhuman Accuracy

Without the ability to correctly generalise from flawed examples, any Machine Learning model trained on imperfect data is bounded to accuracy at most as high as that of the original dataset. This issue can be addressed in a range of ways, from full re-labelling, semi-supervised co-operative labelling, or Active Learning. Fully re-labelling the dataset manually by several experts is time-consuming, but increases the accuracy by increasing the probability of disagreement over incorrect labels. For example, five experts label a dataset of 100 images for a binary label. If the accuracy of every expert is 95%, the joint accuracy of 5 experts under agreement is $1-(1-0.95)^5 = 0.9999997$, or 0.99997 for a dataset of 100. In practice, co-operative labelling is used in an iterative process, where multiple models are trained and manually examined, so that incorrect labels

may be found and analysed by experts, effectively achieving an offline Active Learning scenario over the course of years. This is the case for multiple re-issued and improved datasets, such as labelled Faces in the Wild (LFW) [HRBLM07], VGG [PVZ15] and ImageNet [DDS+09], where state-of-the-art algorithms effectively guide the improvement process by pointing to errors in the original labelling. Finally, active labelling can be used to reach an accuracy higher than that of an expert, by seeking potentially mistaken labels to be scrutinised at training time, as with the Microsoft COCO [LMB+14] dataset, where every image was labelled by a varying number of Amazon Mechanical Turk annotators [CFL+15], depending on interagreement and agreement with a retrained predictive system.

This problem is intrinsically linked to selecting the appropriate oracle for a specific labelling application. This may be done by modelling the accuracy of each oracle for each section of the data, or sub-task, or modality. Similarly to the multi-armed-bandit problem [KVJ87] where the agent with highest-probability payoff is selected online, oracles with highest-probability correct labels are to be addressed for specific tasks.

Superhuman accuracy has already been achieved for graphical tasks such as inpainting and superpixels, but recent work has also achieved superhuman accuracy in facial recognition, OCR[1], and driving.

2.7 Shortcomings and Possible Improvements

Active Learning methods can offer improvements in accuracy given the ability to flexibly label data [KSF17, BRK19]. In the case of human experts, this challenge is rarely addressed, and research focuses instead on supervised and unsupervised approaches on fixed datasets [ABC+16]. This is not the case for Active Learning with an existing expert system, such as labelling by responses from pre-trained systems, or unlabelled retrieval from large data sources, such as image retrieval via online search.

When Active Learning is applied, a challenging tradeoff has to be addressed: system transparency versus accuracy. The state-of-the-art in numerous fields is based on deep learning with neural networks, which is not interpretable, and the network presents no way of explaining the choices it has made. Only limited insight of the inner workings of a trained deep net can be gained, such as low-pass filters [LGRN09] and class shapes [MOT15]. In comparison, other approaches with Bayesian interpretation [SKFH16] and numerical explanations [SMV+19] can be fully transparent, at the expense of prediction accuracy.

Therefore, Active Learning is often used in a collaborative setting: a simpler transparent and interpretable model is used to optimise dataset collection, and a second deep model is then trained on such data, resulting in a high-accuracy result with reduced manual annotation.

[1] Optical Character Recognition

CHAPTER 3

Related Work in Machine Vision

This chapter discusses standard Deep Learning approaches in machine vision, techniques specialised in working on small datasets, and finally Active Learning methods for Computer Vision and Computer Graphics. In order to create an accurate discriminative or generative model, it must have the ability to fit the data, the data must be representative, and the model parameters must be well-tuned. These are three critical tasks in Computer Vision, and while this work focuses on the second aspect (data), the other two aspects must also be taken into consideration. Data is made representative by increasing in quantity, as can be witnessed from better models on better data, and improvements due to model flexibility being directly linked to the availability of large datasets. This demonstrates that the issues of model complexity, model fit, and data are inherently interlinked, and must be assessed in concert.

This chapter lists the objectives and tasks of Computer Vision and how they are approached via Active Learning. The relevant datasets and methods employed in their construction are explained in sections 3.1 and 3.1.1, including labelling applications of Active Learning. Next, classical Machine Vision methods are listed in section 3.2, followed by methods applying Active Learning via Transfer Learning in section 3.3, where Last-layer retraining and Fine-tuning are discussed. Here, section 3.4 demonstrates how Transfer Learning may be applied to produce a baseline for Image Captioning. Next, Generative Adversarial Networks (GAN) and their applications are presented in section 3.5, followed by a section on how labelled and unlabelled data can be utilised 3.6. Section 3.6.1 also discusses pseudolabels and their applications, and shows how these are used to make use of weakly annotated data.

3.1 Dataset Construction

Classical computer vision algorithms relied on manually tuned feature extractors and training of high-dimensional classifiers. Such approaches are less sensitive to dataset size because the number of parameters of typical models is linear with respect to feature size.

With the advent of Deep Learning approaches, the number of parameters in machine vision models has gone from thousands to millions [Kri12, DDS+09, LBBH98a]. Whenever a sufficient amount of data is available, the Deep Learning models outperform traditional approaches [RDS+15a]. Therefore, it is not surprising that the success of deep neural networks relied on two factors: the use of convolutions to reduce the number of parameters [CSVZ14], and the ImageNet dataset [KSH12].

3.1.1 Cannonical Datasets

Deep learning has radically changed the field of Computer Vision, but only thanks to the advent of large datasets. Some of the most relevant datasets are listed in table 3.1, along with the number of categories and number of images per instance. The first of these was ImageNet [DDS+09], created by manually attaching images to 1000 selected WordNet [Mil95] classes. Each image is labelled with exactly one class, with all others implicitly unknown, but practically considered negative. While this dataset has been qualitatively improved, it remains useful mainly for classification tasks, because the location of objects is not annotated. It is important to note that negative annotations are not present, and as images often contain multiple classes as in the Multilabel Learning setting, the correct label at evaluation time is often different from the single label for a given image. Therefore, it is natural to evaluate the dataset with a top-1, top-5, or top-10 score, where the actual label in the data is one of the top-X predicted labels.

These datasets are typically constructed by Class-domain-wise tagging. This process involves the creation of a set of tags/classes/objects, retrieving a large set which is likely to contain them, and manually or semi-manually sifting through the data to produce a large, accurate dataset. However, this approach suffers from various types of bias, which will affect the resulting models' ability to generalise: Sample bias, where the data is representative of the sample, but not the wider population; Reporting bias, where some tags are considered obvious, and others are repeatedly annotated, such as „sky“ or „person“; Selection bias, Group attribution bias, and others [XHE+10].

Annotators may also be influenced by Repetition Priming, where the order and sequence of images presented affect the focus and perception of relevant information to be labelled.

This limitation is addressed by the MS COCO dataset [LMB+14], which contains millions of per-pixel annotations of hundreds of classes. Methods for training detectors and segmentation algorithms on data with no bounding boxes have been demonstrated to work successfully, but an evaluation is impossible without labelled segmentations. Thanks to its size, MS COCO enables per-pixel training for detection and segmentation, thus going well beyond evaluation benefits. MS COCO, therefore, achieves the ambitious task of being an exhaustive full dataset, as opposed to sparsely labelled datasets where only some information is known for any given image.

For a thorough review of image datasets and annotation methods, see [LMB+14].

dataset	# categories	# images containing instance
MNIST [LBBH98b]	10	5 421 - 6 745 (mean 6 000)
ImageNet [RDS+15a]	1 000	732 - 1 300 (mean 1 281)
PASCAL VOC 2012 [EEVG+14]	20	303 - 4 087 (mean 834)
SUN 397 [XHE+10]	397	100 - 2 361 (mean 274)
Caltech 256 [GH07]	256	80 - 827 (mean 119)
Caltech 101 [FFFP07]	101	31 - 800 (mean 90)
MS COCO [LMB+14]	91	~300 - ~600 000 (mean 7 849)

Table 3.1: Comparison of image classification datasets

3.1.2 Active Learning Dataset Construction

Scalable dataset production with Active Learning has been shown to exceed other approaches in terms of quality and scalability as to the amount of laborious and monotonous annotation [CDLFF08]. Object recognition and scene recognition have been shown to benefit from an active learning approach as well [LG13], with new development of loss functions still being researched [SED19, YK19]. For tasks where the amount of required labelled data is prohibitive, such as reinforcement learning for Robot Arm manipulation in response to visual stimuli [SUB96], active learning approaches enable the gathering of relevant data.

MS COCO has been collected in a series of work-intensive steps performed by crowdsourced workers, in addition to validation tasks performed by the authors. There are two core tasks: instance, segmentation and captioning. Instance segmentation was performed in three steps, as described in figure 3.1: labelling selected categories, locating instances of categories, and segmentation of each instance. There are 91 selected categories in 11 groups labelled in the dataset, causing a smaller representativity than ImageNet or others, but with per-pixel segmentation accuracy.

The MSCOCO dataset contains 2 500 000 segmentations, performed by crowdsourced workers over 80 000 hours [LMB+14].

3.1.3 Deep Learning with Small Datasets

New trends in Computer Vision either focus on improving results on established large datasets [KBZ+19, TVDJ20, LVKB19], or on applications in new domains where only small datasets are available [MMK+16, ZZY17]. Data-hungry approaches are pitched against each other on the standard datasets of section 3.1.1, and this work is not focused on improvements on that front. Instead, other methods are being presented which mitigate the need for large datasets, while relying on millions of parameters inherent to Deep Learning [TVDJ20].

Some simple approaches help reduce the need for a large dataset, and these are collectively called image augmentation. Random flipping, cropping, geometric deformations, per-pixel noise, and colour changes are used to create new images with known properties for training, and this has demonstrated benefits on Deep Learning with small and large datasets. Image

Figure 3.1: The MS COCO annotation pipeline, in sequence: (a) Labeling the categories present in the image, (b) locating and marking all instances of the labeled categories, and (c) segmenting each object instance. Taken from [LMB+14]

augmentation is therefore considered the standard, and other methods need to be devised to make deep models practical on datasets so small that image augmentation is not sufficient.

These methods are roughly categorized as follows: weight re-use, simulated data training, and generative models.

3.2 Contemporary Computer Vision methods

This section discusses state-of-the-art discriminative and generative methods for image processing, most of which focus on the model, not the data. Therefore, these are not designed to integrate Active Learning. As will be shown later, judicious application of Active Learning principles systematically significantly improves the results of these approaches, irrespective of their inner mode of operation.

Image classification is an important and challenging problem of Computer Vision. Traditionally, visual categories could be learned by Support Vector Machines on histograms of local features [VDSGS10]. Current approaches have shifted towards Convolutional Neural Networks [KSH12, SEZ+13, DJV+14], which require vast amounts of data and computational power to learn millions of parameters. Such approaches have achieved near-human performance in face recognition [TYRW14], image segmentation [Kok15], and have beaten previous approaches in classification of both very broad and very specific categories [RDS+15a]. The motivation for our approach is to make it possible to use datasets with few examples, but it may also potentially be used to fine-tune Convolutional Neural Networks (CNNs) which already achieve high accuracy.

Convolutional Neural Networks produce state-of-the-art results but deal poorly with small datasets. Class complexity and variability are decisive for defining sufficient dataset size, and we consider any dataset with an insufficient number of examples to be “small”. The pseudolabel method, section 3.6.1, uses an unlabelled dataset to mitigate this.

Convolutional Neural Networks [LBD+89] are the state-of-the-art approach for image classification, achieving the best accuracy for classification and detection [RDS+15a]. These methods require large datasets [TE11], and this is handled by dataset augmentation with rotation, distortion, and other changes to the used images [KSH12].

While much excellent work has been done to enhance the abilities of CNNs on large datasets [WYS+15, SZS+13, SEZ+13, SZ14, SLJ+14, OBLS14], it has generally been accepted that small datasets cannot be directly trained upon with random weight initialization. In this work, we focus on using the CNN structure to improve accuracy, rather than explicitly attempting to improve features, because features can be transferred from classifiers trained on other datasets [YCBL14].

Other approaches to train on small datasets without Neural Networks have been published, with limited success, such as generative models [FFFP07] and a V1-like model [PCD08].

3.3 Transfer Learning

Training a new model for similar tasks, such as the detection of a new object when object detectors are already available, can be time-consuming, and result in lower accuracy. In order to mitigate some of these drawbacks, the following Transfer Learning techniques are widely used: Last layer(s) retraining as described in section 3.3.1, and fine-tuning of section 3.3.2. Applying these has repeatedly demonstrated improvements, thanks in part to the transferable nature of the problems of image processing.

3.3.1 Last Layer retraining

It is well known that training very deep models of 10+ layers using backpropagation is slow and requires vast amounts of data, but results in networks with more representative hidden layers and more accurate models. For example, a network of 100+ layers was the state-of-the-art in image classification in 2017, at the inconvenience of complex training. Training such networks is done by hierarchically adding layers during training, rather than keeping a fixed architecture.

This accruing process is similar to using a well-tuned deep neural network, and retraining the last layer only to suit a task [Gol08, Ben12]. This approach is widely used in Tensorflow /citebtensorflow2015-whitepaper, a deep learning training library used for reference models, in research, and in deployment.

The high-level concept behind the technique is that feature extractors and object-part classifiers are common to most tasks in computer vision, and can, therefore, be re-used across tasks. As shown in figure 3.2, these have been visualized in previous work [YCN+15], and it can be seen that the lower layers are analogous to hand-designed wavelet transforms, and the mid-layers are similar to SIFT and SURF feature extractors.

For example, the reference neural network AlexNet [KSH12] is taken, the last layer is reinitialized and connected to 100 output neurons, and the last layer is retrained to classify to dogs /citebtensorflow2015-whitepaper, with state-of-the-art results. The same approach can be applied to other classes, other tasks, and other problem domains.

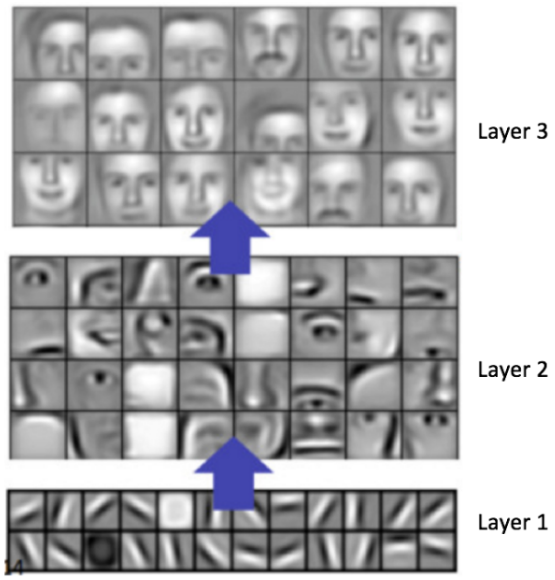


Figure 3.2: Visualized convolutions trained in a three-layer face-detection deep neural network. Taken from [LGRN09]

3.3.2 Fine-tuning

Tasks in computer vision exist, where last-layer retraining is not sufficient in terms of achieved quality, or not usable for other reasons. In these cases, one can turn to fine-tuning, where more extensive changes can be made to the network while maintaining some of the weights of the original network [HOT06, CPH05].

Some instances where this approach may be more appropriate are: when the input size changes, when the input number of channels changes, when the hidden layer size changes due to problem complexity, or when a network is being initialized for an entirely different task [ZLLS19]. In fact, there are benefits to re-using weights even in the last case, because networks spend a large amount of initial computational time training to detect simple features, and this process is sped-up.

A practical example of fine-tuning may be the processing classification of dogs [ABC+16], where a network previously trained to detect various objects must specialize its hidden layers as well as the last layer. Here, fine-tuning is expected to produce more informative channels for dog-specific low-level features, such as eyes and their variations, hair, ears, paws, and tails.

3.4 Image Captioning

The task of generating text which describes the content of an image is called Image Captioning. At the time of publication of my early work on the subject, detecting objects and their relationships was already possible, but generating credible sentences on any topic was in its infancy. Therefore, I presented a baseline approach: Image Captioning with Semantically Similar Images [KHZ15].

assessment	this method	human	random
Ratio of captions evaluated as better or equal to human	0.194	0.638	0.007
Ratio of captions that pass the Turing Test	0.213	0.675	0.020
Average correctness of the captions on a scale 1-5	3.079	4.836	1.084
Average amount of detail of the captions on a scale 1-5	3.482	3.428	3.247
Ratio of captions that are similar to human description	0.154	0.352	0.013

Table 3.2: Scores on the MS COCO Captioning Challenge 2015

In this method, the oracle answers the question: 'How would similar images be captioned?', by finding the per-word count mode and presenting this phrase.

The method uses Convolutional Neural Network activations as an embedding to find semantically similar images. From these images, the most typical caption is selected based on unigram frequencies. This Conformal Predictors approach is used in various other scenarios, and it is a known beneficial Active Learning framework [MSDC12].

Image Captioning is a challenging problem which requires a smart and careful combination of Computer Vision with Natural Language Processing. The approach presented here yields results which outperform several state-of-art published methods while being significantly simpler.

In this approach, the last hidden layer of a Convolutional Neural Network is used as an embedding. For a given test image, the nearest training images are found, and their captions are retrieved. In this body of captions, word counts are used in selecting the sentence which contains the most repeated terms, and this sentence is used to annotate the test image.

This simple method was entered into the Microsoft COCO [LMB+14] 2015 captioning challenge, where it was evaluated by various widely used metrics, and assessed by human judges through Amazon Mechanical Turk. These results are presented and discussed here.

3.4.1 Semantic Similarity Captioning

In order to simplify my approach [KHZ15], it is divided into three steps: CNN embedding (3.4.2), Finding similar images (3.4.3), and caption selection (3.4.4).

For a test image, a pre-trained image classification CNN is evaluated, and the last hidden layer is used as an embedding. In this embedding space, n nearest training images are chosen. All the captions of these training images are then bagged together, unordered. Finally, one of these sentences is selected as the annotation of the test image.

3.4.2 CNN Embedding

The semantic image embedding is computed using the Caffe reference network [JSD+14], pre-trained on ILSVRC [RDS+15b] images. Specifically, the embedding is provided by activations of the last hidden layer after the ReLU nonlinearity. The activations are a sparse vector of length 4096 and were shown to be suitable for semantic content-based image search [CZ14]. This is evaluated on all training and test images.

count	word
83	~~a~~
47	train
21	~~on~~
19	~~the~~
15	down
14	track
13	tracks
11	bus
10	traveling
10	~~is~~
7	~~with~~
7	~~to~~
7	road

Table 3.3: Most frequent unigrams for the example image. Ignored words are crossed.

3.4.3 Finding Similar Images

For a query image, the n nearest database images by cosine distance are selected. See figure 3.3 for an example. n is chosen manually, to fit the task, and $n = 10$ is chosen for the MS COCO 2015 challenge. Other distances were tested (1-norm, 2-norm, ∞-norm, and ranking by linear SVM), but these did not outperform cosine distance.

3.4.4 Caption Selection

All captions of the n most similar images are bagged to create a description corpus – ranks of images are ignored. Since $n = 10$ and 5 captions are given for every database image, a corpus of 50 candidate sentences is obtained (see figure 3.3c). From the candidate sentences, the most representative one is selected by iteratively removing sentences which don't contain words which occur most frequently in the 50 candidate sentences. This culling process starts with the most frequent word.

If a word is not present in any of the remaining candidate sentences, it is skipped. The process ends when only one sentence remains. The 100 most used words according to Google n-grams [MSA+11] are ignored in the culling process. Table 3.3 shows the words used in the example image.

3.4.5 Results

Figure 3.3 shows an example of images retrieved using the CNN embedding. As in this example, the retrieved images match semantically more than visually, as desired. Figure 3.3c shows all candidate sentences of the retrieved images, and Table 3.3 shows the words used to select the final caption.

In the MS COCO Captioning Challenge 2015, resulting captions were assessed by human judges according to five metrics. Table 3.2 presents this method's score for each metric, along with the score for human and random annotation.

(a) Query image (b) Retrieved images

Guy stands near a train carrying gravel on its holding cars
A man standing next to train on a train track.
A train with multiple carts and a person working on it.
A man standing next to a train on train tracks.
A train attendant stands near a train pulling two filled cars.
A train hauling a van is crossing some railroad tracks.
Train passing a man on rural country road.
A train with a man on the back of it with a vehicle in the background
A train car carrying a man and a white van
Much needed train track repairs are now in progress.
A yellow bus carrying passengers riding along the road.
A yellow school bus driving down a street with a red car following behind it.
There is a bus driving down the street.
This yellow bus is driving down the street
A yellow bus traveling down a cobblestone road.
A train driving down the tracks near a platform.
A train that is sitting on a train track.
People on a platform watching a steam train pull in
A train traveling down tracks next to a loading platform.
A train driving toward a station where people are waiting.
A long train traveling through a tree covered countryside.
A multicolored freight train on a track amid greenery.
A train has many containers attached to it
A train traveling down the tracks through scenic scenery.
a long colorful passenger train going down a track by some trees
A train traveling down a train track next to trees.
A train on the tracks moving through bushes.
an image of a train riding along the rail road track
A railroad train traveling down the train tracks
A train is approaching on a railroad track.
A train is riding down the tracks in the middle of some woods.
A very long large train going down a track.
A long train on a track next to another track.
A colorful train sits on the tracks on a foggy day.
A train with two engines pulling cars along the curve of a fall photo.
A locomotive train engine is pulling cars along a railroad track.
A train with passenger cars on train tracks.
A train with an older locomotive drives through the country.
a train sitting on a track next to a bunch of trees
Steam train engine on the tracks in a field.
A train traveling past a building and two lights.
The train is passing by parked trucks in a lot and buildings.
A train that is sitting on the tracks.
A passenger train that is traveling down some tracks.
a big train drives down a track through a city
A bus moving down a road lane designated for buses.
A red and white bus traveling on a side road.
A bus moving fast along an interstate highway.
A red and white bus traveling the bus lane on a highway.
a public transit bus on an empty road

(c) Annotations from selected images, with the selected one in bold

Figure 3.3: Example caption and intermediate results

Although the method received low scores with automated evaluation metrics and in human assessed average correctness, it is competitive in the ratio of captions which pass the Turing test and which are assessed as better or equal to human captions.

3.5 Generative Adversarial Networks

Direct Deep Learning with image input is a discriminative machine learning method, because the inner weights cannot be used to generate new samples from the training data distribution. However, by introducing Generative Adversarial Neural Networks in 2014 [GPAM+14], Ian Goodfellow made a breakthrough that challenges this limit. This new approach combines two networks, a generator and a discriminator, and jointly trains them to produce a generative neural network for new samples from the training set.

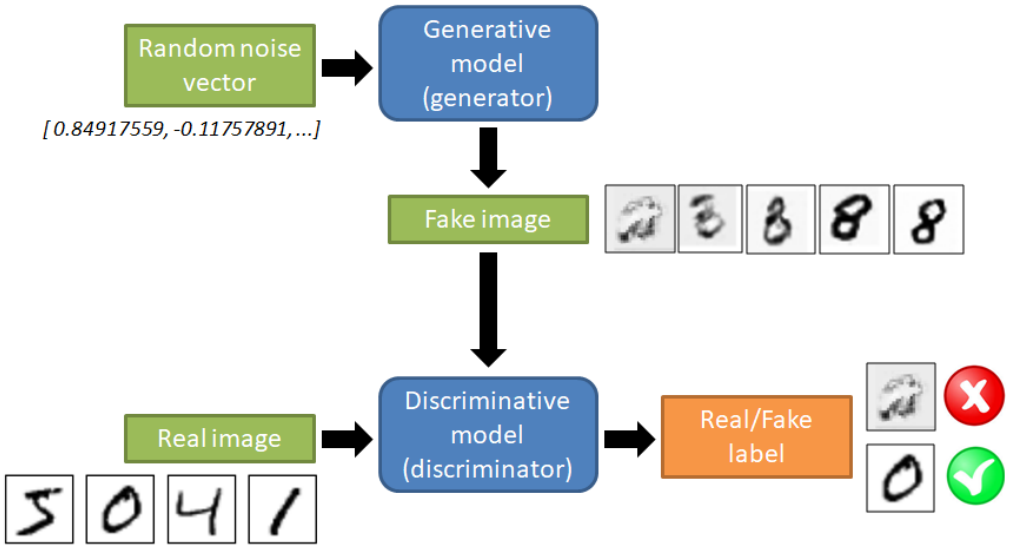


Figure 3.4: Adversarial training of GANs with real and fake images. Taken from [Roo17]

GANs are a form of PU learning, training a generator on data from a set of images with some shared class or property. The generator's input is a vector of random numbers, called the Random noise vector, latent space, or sampling distribution. The generator network consists typically of several deconvolution layers to produce an image output.

This output is passed to the second network, the discriminator. This model takes images, and outputs a single $0-1$ real/fake label. The entire GAN is then trained in two modes:

1. Discriminator training mode, where real and fake images are presented to the discriminator network, and it is trained by backpropagation to distinguish them.

2. Generator training mode, where the two networks are appended, the discriminator network weights are fixed, and updates are channelled via backpropagation from the true/-false output to the generator. The weights are updated accordingly.

The cost of training the discriminator is

$$J^{(D)}(\theta^{(D)},\theta^{(G)}) = -\frac{1}{2}\mathbb{E}_{x\sim p_{data}} \log D^{(x)} - \frac{1}{2}\mathbb{E}_z \log\left(1 - D(G(z))\right) \tag{3.1}$$

where $J^{(D)}$ and $J^{(G)}$ are discriminator and generator costs, $\theta^{(D)}$ and $\theta^{(G)}$ are model parameters, x is every datapoint, and z is every sample from the latent space.

Similarly, the cost of the generator is

$$J^{(G)} = -\frac{1}{2}\mathbb{E}_z \log\left(1 - D(G(z))\right) \tag{3.2}$$

Another variant of GANs are conditional GANs, where the generator input is composed of a random noise vector and a descriptor. Similarly, the descriptor is added to the discriminator output. By using labels from the data, the generator is able to learn to produce random samples with a given property. This technique is described in chapter 5, where it is used twice to train image generators.

While GANs are notoriously challenging to train with stability, they are effectively impossible to overtrain [GPAM+14], because the space being explored is fitting a model to another model, rather than a model to data. If too few samples are used, the resulting generator may be unrepresentative of the real world distribution, but not of the distribution presented to it during training.

3.6 Semi-supervised Learning

As Active Learning proposes relevant datapoints for labeling during training, it is a method of semi-supervised learning, which combines labeled and unlabeled data. While semi-supervised learning does not include annotation feedback, it nonetheless includes important methods used in Computer Vision, especially when labeled data is rare in comparison with unlabeled data.

One such approach is Pseudolabels [Lee13], which supervises the training of a deep Convolutional Neural Network by using a combination of labeled and unlabeled data.

3.6.1 Pseudolabel

Pseudolabel [Lee13] introduced Semi-supervised Learning to Convolutional Neural Networks. As shown in Algorithm 1, the CNN is trained in the usual way, but training images are supplemented by an unlabeled dataset. A low-density separation between classes justifies the use of entropy regularization on additional data.

Data: labeled images, unlabeled images
Result: trained classifier
initial training of CNN with labeled images only;
while *CNN not converged* **do**
  for *each unlabeled image I* **do**
    pick the class with max predicted probability
  end
  train CNN with labelled and weighted pseudolabeled images
end

Algorithm 1: Original pseudolabel algorithm

In addition, at each iteration, the mixed set is classified with the current network, and these predictions are used as labels for the next iteration. Random selection from the mixed set,

and increasing weights for the selected subset, are meant to help convergence to a classifier principally influenced by the training set.

This approach is justified by the cluster assumption, which states that the decision boundary should lie in low-density regions to improve generalization performance [CZ]. Rather than explicitly searching for low-density regions, the pseudolabel approach implicitly finds these, because changes in classification are more likely to occur in regions where the consensus among examples can be perturbed by few label changes. The pseudolabel approach helps with the MNIST dataset, divided artificially into a training set and a mixed set for which labels are unknown. Accuracy comparable to that achieved by using the entire set was reached. However, this dataset is long considered solved [WZZ+13], and similar results remain to be demonstrated on a challenging problem.

3.7 Summary

Machine Vision has seen a rapid improvement thanks to the application of highly flexible models with millions of parameters, trained from the ground up, and thanks to datasets several orders of magnitude larger than before. While datasets have been gaining in size and effect on model quality, so has the need for extensive manual labelling [ZLLS19]. After a careful Class-domain-wise tagging initialization, canonical Image Datasets used to be first annotated by domain experts, then by groups of researchers, and now require tens of thousands of work-hours by a large body of crowdsourced workers [DDS+09, Kri12]. In order to best utilize each of these, and all three types of workers in unison, Active Learning is becoming the reference method of dataset construction [YSZ+15].

Automatic and semi-automatic annotation is necessary for training next-generation classifiers, detectors, and segmentation methods, because of the exponential growth of necessary data. Therefore, this thesis now presents a systematic analysis of the question of how to best apply Active Learning to Machine Vision.

CHAPTER 4

Hypothesis & Contribution

Generative and discriminative algorithms in Computer Vision are designed and trained to maximize their ability to generalize. This is tested on unseen data, and maximized by improving the model prior, improving the quality of the labelled and unlabeled data used, and by hyperparameter tuning. A tangential approach, compatible with improving the model and hyperparameters, is Active Learning, by which the gathered data is improved during training-time by effectively managing the labelling and selection effort of an expert.

4.1 Hypothesis

As it is desirable to optimally transfer knowledge and existing models to new tasks where data is limited, the following hypothesis is put forward:

> *Human and algorithmic expert annotation using Active Learning improves the accuracy of contemporary Computer Vision methods.*

The hypothesis is put forward with the expected result of achieving a significant margin, thus being both useful and demonstrably achievable. The term *contemporary* is taken to mean algorithms performing state-of-the-art accuracy in the case of discriminative models, or output quality for generative models, but not expecting to produce comparable results to next-generation models. The hypothesis refers to *human and algorithmic experts* because Active Learning is expected to be applicable both in the context of green-field datasets and applications and in Transfer Learning applications where one or more useful weak learners already exist. The concepts of *accuracy*, *quality*, and *labor time* are context-specific, referring to a context-dependent relevant applicable metric of generalization, and to the time and effort taken to achieve said results.

4.2 Method

The hypothesis is tested by comparing the achieved accuracy of this approach, as opposed to hyperparameter tuning, model tuning, and extended training time. These alternative approaches are known to hold the potential to improve accuracy and reduce labour time for Machine Vision tasks, and they are widely studied elsewhere [DDS+09, YRLT, TFF08, DBLFF10, AQ07, SGZ+16, RMC16, LBBH98a, KLA+20]. The experimental proofs presented further are divided by approach and refer to my own relevant published work. These are experiments to demonstrate the benefit of Active Learning via sample selection, via human and algorithmic labelling, via feature selection and adaptive visualization, and pseudolabels.

The standard supervised, unsupervised, or semi-supervised setting can be formulated as training a model M with parameters ρ on two sets of data:

$$\rho = \underset{\rho}{\operatorname{argmin}}\, M(X_K \cup X_U) \tag{4.1}$$

Where X_K are datapoints with known labels, and X_U datapoints with unknown labels. The Active Learning approach has the additional $X_{C,i}$ subset of X_U, which contains the datapoints chosen to be labeled:

$$\begin{aligned} X_{C,i} &= S_i(M, \rho_i, X_{K,i}, X_{U,i}) \\ \rho_i &= \underset{\rho}{\operatorname{argmin}}\, M(X_{K,i} \cup X_{U,i} \cup X_{C,i}) \end{aligned} \tag{4.2}$$

And the hypothesis is therefore simply that there exists a selection mechanism S such that

$$\forall i \, \exists S \quad | \quad L(M(X_K \cup X_U)) < L(M(X_{K,i} \cup X_{U,i} \cup X_{C,i})) \tag{4.3}$$

Where L is an appropriately chosen loss function for the given model.

The results of the demonstration of the hypothesis have a wide array of applications, such as Association-rule Learning used by comparing the quality of texture synthesis algorithms over inputs with selected properties [KDC17, KCD15].

In the following chapter, I will test the hypothesis in various scenarios. These tests correspond to various perspectives under which Active Learning can be utilised, namely online training with an optimized graphical interface [HKL+12] in section 5.3, iterative dataset optimisation [KHZ20] in section 5.1, and active transfer learning with an algorithmic expert [KHZ16] in section 5.4.

4.3 Contribution and Proof Outline

The theoretical contribution is that these results irrevocably demonstrate that the systematic application of Active Learning improves the accuracy of contemporary Computer Vision models. A practical contribution is also made, in the form of an algorithmic process usable for Zero-shot Learning and One-shot Learning for image classification, given that a weak retrieval system is available from a large set, such as online image search indexing the Web.

Furthermore, applied contributions are made in the fields where the hypothesis has been tested: an improved dataset and algorithm for Font Capture and a GUI for image tagging with Active Learning.

Specifically, the existence claim of equation 4.3 is demonstrated by finding S for various scenarios of (L, M, X). For each, the contemporary Machine Vision approach is considered in comparison with the Active Learning alternative. These two options are then compared, with the desired objective of showing

$$\exists S \quad | \quad (L, M, X) \tag{4.4}$$

under the conditions of equation 4.3. By demonstrating this for four important problems of current research, the hypothesis is validated in a limited context. In order to validate the hypothesis with respect to all problems of Computer Vision, the argument by analogy is made that every Computer Vision problem can be aided by applying these Active Learning principles.

The cases under consideration are:

1. Tagging of closed-domain information by optimizing an Active Learning Graphical User Interface
2. One-shot learning with pseudolabels and a weak algorithmic expert
3. Active Learning for human experts to create data for Generative Adversarial Neural networks
4. Transfer Learning of algorithmic experts for Generative Adversarial Neural networks

These four cases correspond to sections of the following chapter, where they are treated in more detail. An overview of the interconnections of these sections and how they jointly support the thesis is as follows: Active Learning benefits the model and users directly by allowing them to more efficiently label data with general classes, as well as domain-specific information (1. - published in 2012). Moreover, sufficiently well-preprocessed image data allows high-quality training of classes without any human expert, by using an online image search algorithm and pseudolabels to train a Convolutional Neural Network (2. - published 2016). The filtering of existing datasets by humans (3. - under review), as well as the labelling of unsupervised datasets by algorithms (4.) can be performed with Active Learning, enabling improved generative quality as well as entirely new applications.

Therefore, these four problems present a holistic approach to the application of Active Learning in contemporary Computer Vision. Beyond the contribution made to the hypothesis, these have also served to further the fields of research they have been applied in, as detailed in the following chapter.

CHAPTER 5

Active Learning for Machine Vision

The cases in which Active Learning has been tested to support the hypothesis are described in this chapter. The presented work is divided into four sections, loosely corresponding to my own published work. By using a combination of Uncertainty Sampling and Diversity Sampling, the sections below focus on creating improved data and models by sampling from all datapoints while optimizing labelling. Each of the following four cases is an experiment to test the hypothesis, and thus to serve as a quantitative proof.

In the context of labelled image datasets, image-wise tagging is not limited to pre-training annotation. In fact, the required expert input can be reduced by judicious initialization with an external system [KHZ16], by asking the annotator to verify rather than label [DDS+09], and by in-the-loop training to identify samples with low certainty [HKL+12].

This chapter describes my own work, in which the first approach has been tested and published, as detailed in section 5.4, the second approach has been tested in the context of generating fonts 5.1, and the last approach has been experimentally validated through implementation and user experiments 5.3. Similarly to the first approach, work in section 5.2 also shows that beneficial results can be achieved with pure Transfer Learning, where a set of labels is created from specialized pre-trained models, serving new tasks not possible before. Section 5.2 presents work made public as a freely available dataset at `https://github.com/DCGM/ffhq-features-dataset`.

These three of sections correspond to peer-reviewed work, as follows: Section 5.1 contains work currently under review at The Visual Computer as *Font Capture in the Wild* [KHZ20], section 5.3 was published as *Annotating images with suggestions—user study of a tagging system* [HKL+12], and section 5.4 describes the method published as *Deep learning on small datasets using online image search* [KHZ16].

Finally, section 5.5 summarizes how these individual contributions support the hypothesis of the thesis, and integrates the findings into a cohesive methodical validation.

5.1 Improvements via Dataset Size and Quality

Font Capture is a task in Computer Vision and Computer Graphics, in which text present in an image is replaced with new text in the same font. Worldwide, 750 million people are native speakers of a language written in a Latin-derived alphabet with diacritics such as accents, subscripts, and superscripts [wik18]. However, out of an estimated 100 thousand digital fonts widely available, only a few hundred include these non-English characters.

Font extraction on characters of the Latin alphabet has been attempted before, either with limited applications to classical fonts [CK14], or with blurry or noisy results [Bal17, USB16], and always by using individual characters as input. Thanks to an improved dataset and method for generating training samples, this work creates sharp fonts extracted directly from a line of text, suitable for use in photo editing as well as vectorization. This approach makes it possible to take an existing TrueType font, render new characters, convert them to vector graphics, and incorporate them in the original, thus effectively closing the loop. Active Learning has been used to create a large high accuracy dataset of fonts, thus improving the quality of the method.

Generating fonts cannot be replaced by font search over a large enough dataset, as shown. Although fonts are widely shared on the internet, and font search engines are freely available, few fonts can be acquired to perfectly match the desired input. Finding a font given an image is a challenging task, undertaken by domain experts or automated processes. Identification methods range from pixel differences on detected aligned characters [squ17] to matching manually entered detailed features [ide17] based on standard font classification techniques [CGL13], or automatically extracted attributes [OLAH14]. If these methods fail, fonts can be identified by a community of experts, such as Fontid.co. However, exotic fonts may be unknown to experts, unavailable to identification systems, or non-digitized. For example, Figure 5.1 shows a query text, along with nearest retrieved fonts by existing methods. This demonstrates that pixel difference and others are not a sufficient metric in font style matching.

These limits of finding existing fonts sparked an interest in extrapolating the entire style of a font from a single example. Font extrapolation with warp mappings dates to the nineties [TF97], inspired by the effect on the shape of charge on ink particles. A manifold over fonts has allowed smooth traversal of the font space [CK14] and was applied to classical typefaces to interpolate fonts. Extrapolation of numerals on the MNIST and SVHN datasets was made possible by deep generative models, creating a latent space which allows traversal across glyphs [KMRW14].

More recently, a fully connected deep net has been used to create an embedding of 50 thousand fonts [Ber16]. A feed-forward neural network has been used to generate the entire font from four characters [Bal17], with poor quality results. In addition to limited quality, this technique suffers from requiring specific characters, which may not occur in the sample text. Variational Autoencoders have been used to generate fonts from a single example glyph [USB16], but with a small dataset of 1'839 fonts and a fully connected network, the results are still blurry. The 50k fonts dataset [Ber16] has been used to train a VAE and a GAN [gan16], using the principles outlined in [RMC16]. Fonts are extrapolated from varying characters with a Multi-Content GAN [AFK+18], in colour. However, existing methods require segmented characters

(a) Query text from image - hand-drawn

(b) Nearest match by pixel difference - **JollyGood Sans Condensed**

(c) Nearest match by property matching - **Keynote** (caron unavailable)

(d) Nearest match by expert community - **Krinkes**

Figure 5.1: Comparison of font retrieval methods

rather than analyzing text directly. Most crucially, results of all existing methods are blurry or noisy for all but the most standard fonts.

While the existing methods train various architectures of neural networks with millions of parameters, I anticipate that increased quality may be reached through the application of Active Learning to create a larger, more representative dataset on which similar methods may be trained. The dataset was made by assembling a large pool of .ttf font files, iteratively training and annotating data for a binary classifier of usability, and thresholding the ensuing classifier to produce a dataset of usable quality fonts. This procedure utilizes a combination of Uncertainty Sampling and Diversity sampling, by focusing the annotator's attention on cases with high certainty, as well as cases of very low certainty.

The dataset is filtered through a shallow Convolutional Neural Network over three iterations. At each iteration, four representative characters of every font in the unlabeled dataset are rendered, classified, some are annotated, and the process repeats. The representative characters are „a“, „l“, „1“, and „?“.

The initialization proceeds as follows: The representative characters are rendered for all fonts and placed into a single image named with the unique font ID. If any of the characters „a-z“, „A-Z“, and „0-9“ is blank or undefined (rendered as in figure 5.2), the font is discarded immediately. Similarly, if any two characters are equal pixel-for-pixel, the font is discarded. All remaining fonts are viewed in a directory, allowing quick preview and easy group selection.

Figure 5.2: Undefined characters render as Unicode error codes

The fonts which do not contain readable Latin characters are manually selected and labelled as negative. This is performed for 0.5% of the data, or 1 300 fonts, which requires about two hours of annotation time. The other seen examples are marked as positive. This annotated data is then used to train a shallow Convolutional Neural Network. The network, used to classify usable fonts on four characters of fixed size, has two convolutional layers of 8 and 2 channels, and a last dense layer with a sigmoid activation function. This simple network is trained on the annotated data. Negatively annotated fonts include non-Latin fonts, dingbats, emojis, and highly ornamental typefaces, which may produce unexpected characters for standard glyphs.

Then, the network is used to make predictions on the unlabeled data. The 99.5% of unlabelled data receives ratings from 0 to 1, for which two tasks are semi-manually performed: the establishment of a threshold τ_1 where positive examples outweigh the negative, and manual labelling of unlabeled fonts near this threshold (uncertainty sampling), and near the 0 and 1 ratings (diversity sampling).

This for of uncertainty sampling is very effective, producing a high percentage of samples to be re-labelled. On the other hand, this simplified form of diversity sampling does not produce many examples to be re-labelled. This can be interpreted in two ways:

1. The classifier is very effective and has few high-certainty incorrect cases
2. This diversity sampling method is not effective at finding new types of cases needing re-labeling

After three iterations, the re-labelled fonts are once again thresholded with τ_3, and the effectivity of the combination of the sampling methods is evaluated as follows. A random sample of positive fonts is taken and manually evaluated until a false-positive is present (an incorrect font selected as correct). Using this method, the first false positive in random data was found at position 349, giving an expected accuracy of over 99.3%.

The dataset is then processed further, to create a specialized section of fonts with diacritical marks. This process is performed as in the initialization stage of the full unlabeled dataset, but over a different set of representative characters: „á“, „č“, „Ď“, „ç“, „Å“, „å“, „è“, and „Í“. If any of these characters were blank, undefined, or initialized with an error Unicode as in figure 5.2, the font was not selected. Upon manual assessment of the quality of this data, it was judged that no further Active Learning was necessary to improve the quality of this portion of the dataset with diacriticals.

In summary, fonts used in this method have been acquired online, with 222 462 used out of 272 849 unique fonts, including 7 089 fonts with selected diacritical marks (an acute accent ´, circle ˚, or caron ˇ on eight characters). The fonts have been downloaded from various sources, such as multiple unofficial datasets, Open Source libraries of fonts and font families, and official repositories of font-sharing websites. A font family is typically a group of related fonts which

vary only in weight, orientation, width, etc., so in order to create a highly representative dataset, it is desirable to include fonts with similar variations. Downloaded fonts have been filtered with the deep net described earlier.

A Generative Adversarial Network was trained on this dataset to render any of the characters with and without diacriticals. While rendering data for training, the input was rendered as ordinary text with correct kerning and English letter statistics, by sampling phrases from a Harry Potter book. The GAN was simultaneously trained to generate diacritics, by using non-diacritics at the input with fonts containing diacritics, and a random accented character at the output. An outline of the trained GAN can be seen in figure 5.3, and further details on this standard process can be seen in the original publication [KHZ20].

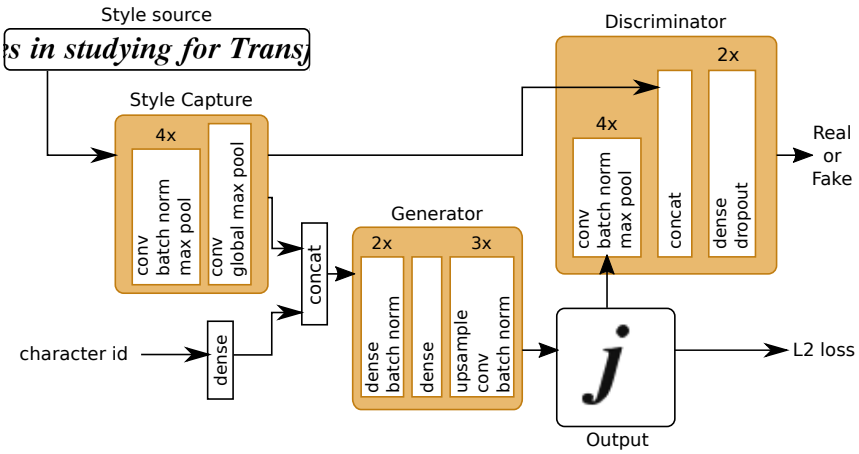


Figure 5.3: GAN structure, with sample inputs and outputs

5.1.1 User Study

A user study with 17 participants compared generated characters from state-of-the-art methods: VAE [KMRW14], ADV-VAE [USB16], and this work. The study was performed with three triplets of characters, as shown in Figure 5.4. Each participant received 72 rows of triplets, printed on four sheets, and was asked to identify the different triplet. If the user fails to identify the generated triplet, the output of the method can be considered indistinguishable from the original font. Correct and incorrect user classifications are summed for each method, and results are presented in Table 5.1. The proposed method recreates fonts convincingly in 51% of cases, compared to 3% and 9% for the previous methods. According to the randomization permutation test, these results are highly significant ($p<0.0001$). Furthermore, tests show that VAE outperforms ADV-VAE with p-value 0.059.

Using this approach, fonts may be extended to other alphabets and non-alphabetic languages for the benefit of billions of people whose native languages are not written in Latin alphabets. Unicode defines 136 900 characters [The11], all of which can be generated in any font using this approach. Office suites such as Microsoft Office & Google Docs can benefit from incorporating such tools in the future.

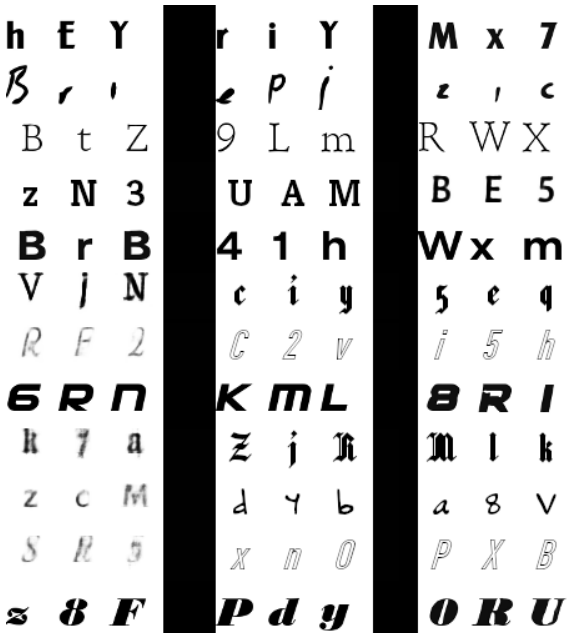


Figure 5.4: User Study Setup. Each row contains three triplets, two of which are ground truth, and one is generated by one of the three methods. Their order is randomized, except for the middle triplet, which is always the ground truth.

Method	success rate	adjusted success rate
VAE [KMRW14]	4.7%	9.4%
ADV-VAE [USB16]	1.6%	3.1%
Ours	25.6%	51%

Table 5.1: User Study Results. If the method produces indistinguishable outputs with respect to the ground truth, the user performs a random binary choice. This corresponds to a 100% method success rate for an ideal output, versus 50% measured in the experiment. The adjusted success rate is doubled accordingly.

A limitation of this work is the lack of kerning information. Currently, kerning is being done manually, so automated letter spacing is only possible for monospaced fonts, but this can be incorporated as an additional specialized task.

The hypothesis has been tested by comparing similar generative neural networks under similar conditions, but the network shown here has been trained with an improved larger and higher quality datasets thanks to the application of Active Learning. The method trained on the dataset produced via Active Learning has demonstrably outperformed the other, as shown in table 5.1.

Figure 5.5: A sample from the Flickr Faces HQ dataset (FFHQ)[KLA19]

5.2 Unsupervised Active Learning

Another practical application of Active Learning principles is Transfer Learning, enabling effective algorithmic annotation of unlabeled data. The Flick Faces HQ (FFHQ) dataset [KLA19] has been successfully used in generating faces by training GANs, but the images lack annotation, which would provide useful information in guiding face generation via features. Due to this limitation, faces can only be generated by randomly sampling the latent space, or by selecting latent space points corresponding to known faces. However, it has been previously impossible to generate random faces with specific attributes.

The FFHQ dataset contains 70 000 unlabeled unique faces in high resolution, making it well suited for applications in graphics. See figure 5.5 for a random sample of these images. These images, even when annotated, would be impractical for training other tasks, such as gender recognition or orientation, because of the unnecessarily high definition and low sample count.

However, by combining state-of-the-art feature extractors with the high-resolution dataset, it is possible to create a dataset of labelled faces with useful information for guided generation of faces with specific features.

5.2.1 Transferring features

The dataset was created by running these pre-trained models to extract features: VGG-Face [PVZ15], Facenet [SKP15], OpenFace [BZLM18], and DeepFace [TYRW14]. These pre-trained models detect faces and then produce features. These are geometrical features (landmarks and orientation), and well as categorical features (facial hair, emotion, eye colour, etc.). The 70 000 faces in were processed by these four models, and all but 528 very detected and annotated. The remaining faces are excluded from additional training in order to maintain the system's full autonomy. Transfer learning from pre-trained experts enables the creation of data labels by applying other algorithms, without human labels or supervision.

Figure 5.6: A random face from FFHQ dataset, with some extracted annotations in .json format

See figure 5.6 for a random annotated face. The resulting dataset, *FFHQ-features*, is available online[1].

5.2.2 Generative Faces with Features

The dataset was used to train a conditional Generative Adversarial Network. Unlike traditional GANs, which randomly sample the latent space to generate samples indistinguishable from training data without control over the features, conditional GANs are trained with an additional control vector, allowing them to set the desired properties of the output, given that this information was known during training.

Several architectures of cGANs were trained, with a number of different tunings, loss functions, and hyperparameters [Ven20]. The features used were only age and gender, but the process can be applied to any categorical and continuous features present in the *FFHQ-features* dataset. Figure 5.7 shows randomly generated faces with gender and age control.

Instead of randomly sampling the latent space, and thus generating new faces, the network can also be used to generate the same face with different control features. In figure 5.8, the same random vector in latent space is rendered with different ages, resulting in the generated ageing process for a random, non-existent person.

By extending the use of Active Learning methods to fully algorithmic solutions via labelling with pre-trained networks, new results have been achieved. It was previously impossible to generate faces with specific features due to the lack of such quality data, and by applying

[1] https://github.com/DCGM/ffhq-features-dataset

Figure 5.7: ProGAN-generated faces with age and gender restrictions. Rows alternate genders, columns hold incremental age groups [Ven20]

Figure 5.8: ProGAN-generated faces, with varying age parameters [Ven20]

these techniques, the contemporary Machine Vision problem of generating faces has been quantitatively furthered.

5.3 Optimising user annotations

In this typical case of Adaptive Learning, a tagging system is presented such that it optimizes the annotation process with respect to two criteria: optimal adaptive recommendations based on prior actions, and an efficient interface for large-scale annotation. However, unlike Collaborative Filtering, this Active Learning use-case focuses on user-specific information, as opposed to preferences on globally known objects.

Most generally, users are presented with the option of creating arbitrary tags and aligning them to their own images. As these can be user-specific, language-specific, or location-specific, the information is not necessarily known for other users and other objects, and every tag has to be predicted online based on current tags.

By using Restricted Boltzmann Machines to provide labelling recommendations in a web-based user interface, the annotation of images via Active Learning has been experimentally tested in a user study. The objective of the tag suggestion methods is to allow Image-wise tagging (assign tags to an image) rather than Class-domain-wise tagging (assign images to a tag). These results demonstrate that large datasets with semantic labels (such as in TRECVID Semantic Indexing) can be annotated much more efficiently with the proposed approach than with current class-domain-wise methods, and produce higher quality data.

5.3.1 Local Tag Suggestion

A Restricted Boltzmann Machine is used to predict labels, by the encoding of the labels of surrounding tags and extracted features. Aside from the RBM suggestion method, tags are also suggested if they are positively annotated in nearby images in the gallery. A gallery is viewed as a chronological sequence, with images $\{I_i\}_{i=1}^{N}$. When generating suggestions for a given image I_i, each tag is given a weight ω, given by

$$\omega = \sum_{i=1}^{N} \frac{1}{log(|p-i|+1)} * has\_tag(I_i), \tag{5.1}$$

where

$$has\_tag(I_i) = \begin{cases} 1 & \text{if the tag is positively annotated on } I_i \\ -1 & \text{if the tag is negatively annotated on } I_i \\ 0 & \text{if the tag is not annotated on } I_i \end{cases}$$

The $\frac{1}{log(|p-i|+1)}$ term ensures that closer annotations have more weight on ω, and the $has\_tag(I_i)$ term ensures that positive annotations have positive weight, negative annotations negative weight, and all others are ignored. Tags are then ordered by their ω from highest to lowest. Any tags with $\omega > 0$ are then suggested, in this order.

Figure 5.9: Typical view of the ITS web interface. Annotation option parts are outlined in red.

5.3.2 Integration of Suggestion and User Interface

When suggesting n tags, $\lfloor n/2 \rfloor$ are from the RBM model, $\lfloor n/2 \rfloor$ from local tag suggestion, and if n is odd, the remaining one is chosen with either method with equal probability. That ensures that when only one tag is being added, neither method is favoured.

When an image is loaded, 15 tags are chosen, and three annotating options are available to the user. As seen in Figure (picture of the Web), they are as follows:

1. Each of the 15 suggested tags is presented with a „check“ and a „cross“. When clicking check, the tag is added as positive annotation, and the cross adds negative annotation. When clicking either, the tag disappears from the suggestion list, and a new one is added at the end of the list.

2. The user can use an auto-completing text field, where any typed word or part of a word is matched with all occurrences in existing tags as a substring. For example, when typing „person“, the user is presented with „person“, „male person“, „female person“, and others. This ensures that when no information is given yet, the user can easily add information that's compatible with the current collection of tags in the database. When any of these is clicked, it gets added to the current suggestion, and the suggested tags are refreshed accordingly. Users are allowed to enter new tags which are not yet in the database; however, such tags are not immediately considered by the RBM model. It is more appropriate to

add new tags to the RBM model when the number of positive annotations of such tags increases over a certain threshold in order to prevent saturating the model by rare or otherwise irrelevant tags.

3. Given the chronological sequence of images, three preceding and three succeeding images are shown on the right. When any of these is clicked, the positive tags that have been annotated on that image are copied over to the current image, and the suggested tags are refreshed accordingly.

The suggestion operation takes on average 0.1 seconds, making the system responsive and allowing quick interaction with the user. In case of sequential video frames, this interface allows users to seamlessly copy tags from previous images to the current one, either by copying tags from the three preceding and three succeeding images or by selecting the suggested local tags.

Another use scenario is the annotation of holiday photos with recurring themes, people, and elements. In the case of unusual images and tags that are not a priory likely, the RBM suggestions may not be accurate very useful at first; however, by providing one or several tags relevant to the image (e.g. by using the auto-completing text field) will make co-occurring tags likely to be suggested.

5.3.3 Experiments and Results

In order to identify the usability and usefulness of this system, two experiments with users were performed: testing with untrained individuals with minimal support, and testing with expert annotators for an extended period of time. In order to make the test replicable, only images and tags[2] from the TRECVID 2011 Semantic Indexing task[3] were used, and the feature to add new tags was disabled.

Besides the reproducibility of the experiments by others, there are several other advantages of using the TRECVID data. A part of the data is already annotated and can be used to learn the RBM tag-dependency model. Further, the dataset was annotated manually [AQ07], which provides a baseline for comparison.

In addition to the user study, the ability of RBM to model dependencies among tags and the ability to estimate marginal tag probabilities by Gibbs sampling was tested on the TRECVID data. This experiment gives the objective information from the RBM suggestion system alone.

Testing by Untrained Users

Ten randomly selected technical university students were asked to use four different tag suggestion methods using this system, with as little training as possible. The four methods are:

1. **none** — no suggestion method

[2] Examples of the classes are Actor, Airplane Flying, Bicycling, Canoe, Doorway, Ground Vehicles, Stadium, Tennis, Armed Person, Door Opening, George Bush, Military Buildings, Researcher, Synthetic Images, Underwater and Violent Action.

[3] `http://www-nlpir.nist.gov/projects/tv2011/tv2011.html`

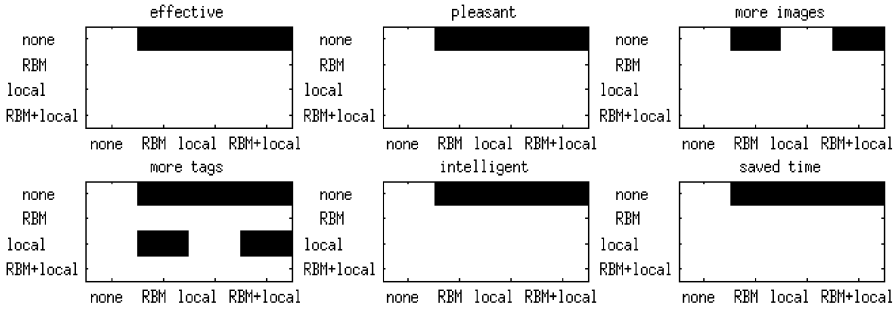


Figure 5.10: Black squares represent a significantly better outcome in the user evaluation, according to the questionnaire. The questions allowed a $1-5$ rating on effectiveness, pleasantness, amount of images, amount of tags per image, perceived method intelligence, and whether the method saved time.

2. **RBM** — only Restricted Boltzmann Machine suggestion (Section 2.4)
3. **local** — only local tag suggestion (Section 5.3.1)
4. **RBM+local** — the combination of Restricted Bolzmann Machine and local tag suggestion, as presented in section 5.3.2

The methods were ordered randomly, and the user was not told which is which. After using each method, the user was asked to answer a questionnaire with questions regarding the rating and usability of the method, and data regarding the number of annotations created was stored.

According to the results (Figure 5.11), **RBM** and **RBM+local** suggestion methods allow significantly[4] faster annotation. There were no significant differences between **RBM** and **RBM+local**, nor between **none** and **local**. According to the questionnaire, method **none** is found by the users to be significantly[5] inferior to all the other methods in almost all aspects. No other significant differences were found, except that **RBM** and **RBM+local** received better marks in the ability to facilitate annotating more tags per image compared to **local**.

Testing by Expert Users

Three expert users were asked to use the combined tag suggestion method (Section 5.3.2). The users previously took part in TRECVID 2011 collaborative annotations [AQ08] and had at least two hours experience with ITS. The users spent a total of three hours annotating randomly selected videos from the TRECVID dataset.

In this setting, the number of positive and negative annotations assigned per hour was 448 and 3085, respectively, averaging 13.1 positive annotations per image. The annotating speed compares very favorably to Class-domain-wise tagging annotation for which the authors of [AQ08] expect 2 seconds per annotation; moreover, only 2.5% of the annotations in the TRECVID 2011 SIN [OAM+11] dataset are positive. When compared to the original distribution of

[4] Using the paired t-test at the 10% significance level.
[5] Using the Mann-Whitney U test at the 10% significance level.

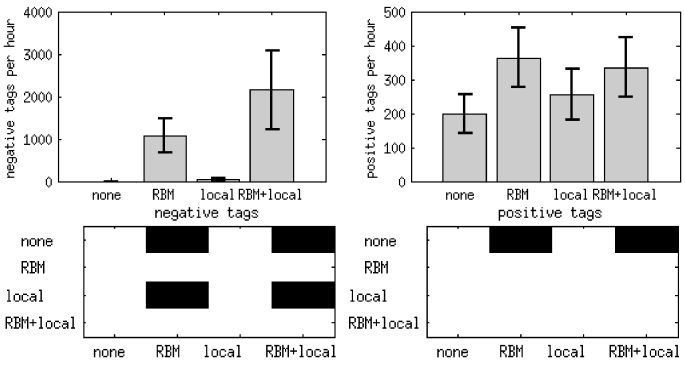


Figure 5.11: The top graphs show the mean number of tags assigned per hour with confidence intervals at 90% significance level. The bottom graphs show black squares where the column methods annotate significantly more tags per hour than the row methods.

tags obtained by the Active Learning method [AQ08], the ITS tags have a heavier tail distribution for both positive (kurtosis 8.35 in TRECVID and 4.18 by ITS), and negative annotations (kurtosis 2.18 in TRECVID and 1.98 by ITS).

It has been shown that the presented method produces higher quality annotations in less time than comparable methods. Therefore, these results support the claim that Active Learning presents an improvement over approaches without it and that the creation of labelled datasets will benefit from the approach presented here.

5.4 Deep Learning on Small Datasets using Online Image Search

Learning image tags and object detectors is a core task of Computer Vision, and the large amount of data required to train every visual class is prohibitive. Therefore, by reformulating the problem in a Weakly Supervised PU learning setting, image categories can instead be trained from algorithmically preprocessed noisy online data. The following approach, the core contribution of this thesis, was presented at SCCG 2016 [KHZ16].

The proposed algorithm utilizes Google Image Search in a Hybrid Action Learning, where active learning with a weak algorithmic expert is used after an unsupervised initialization. Thousands of images are retrieved for any search string. The resulting set of images is weakly annotated, in the sense that numerous examples may be wrong or noisy. The data is stored statically for each given class, so this is not presented as a Online Learning problem but as an Incremental Learning problem.

The proposed algorithm (Algorithm 2) is composed of an initial pre-training, a selection process, and a repeated weighted training step.

This section describes the data, the method, and the implementation.

In the original paper [Lee13], pseudolabels are labels assigned during each epoch to any unlabeled images based on classifier responses. In the current setting, pseudolabels are weighted labels of the class used to query each image in online image search.

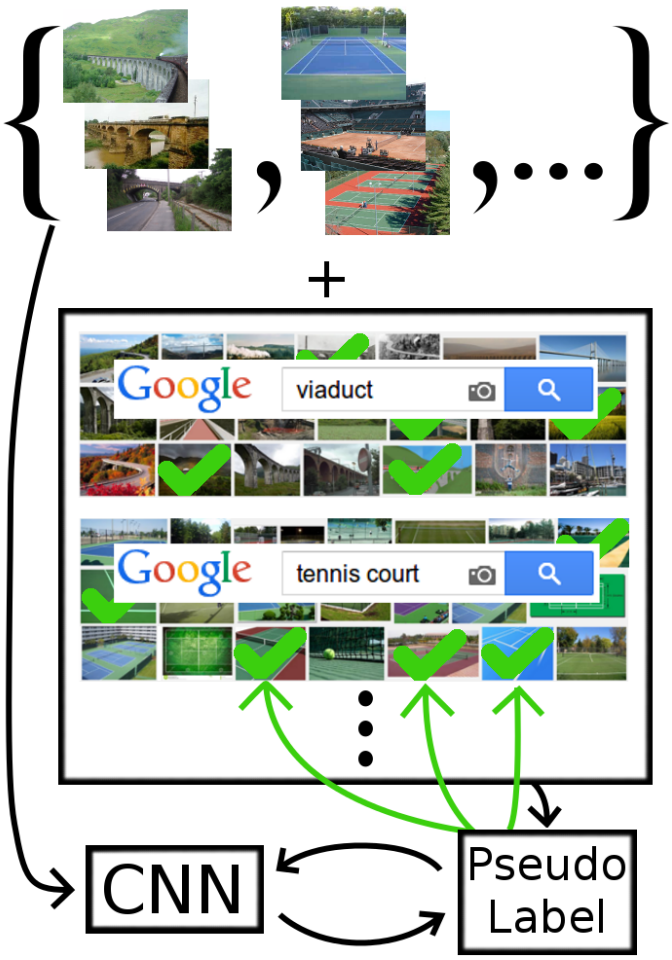


Figure 5.12: Pseudolabel selects useful additional images from an unreliable source, to help train a Deep Learning classifier

Throughout this section, the following conventions are adopted: $\mathbf{X}$ is a set of images $\{X_1, X_2, X_3, \ldots\}$, $\mathbf{y}$ is a set of labels $\{y_1, y_2, y_3, \ldots\}$ where $y_n \in \{1, 2, \ldots C\}$. C denotes the number of categories. Training examples have the form $(\mathbf{X}, \mathbf{y})$. Every i model update iterations is referred to as one epoch, and a set of images and labels during the duration of epoch e is denoted $(\mathbf{X}_e, \mathbf{y}_e)$.

labeled images are divided into a training set and testing set: $(\mathbf{X}^{\text{train}}, \mathbf{y}^{\text{train}})$, $(\mathbf{X}^{\text{test}}, \mathbf{y}^{\text{test}})$.

In addition to the train and test sets, query images are retrieved from an online image search engine separately for each category. The queried images are denoted $(\mathbf{X}^{\text{query}}, \mathbf{y}^{\text{query}})$.

5.4.1 Training CNN

CNNs are trained by Stochastic Gradient Descent, where training images are propagated forward through the network in batches to produce outputs, for which error gradients are calculated. To complete an iteration, these are backpropagated to calculate loss gradients, which are used to update network weights. This process is repeated until convergence.

Data: labeled images, queried images for each class
Result: trained classifier
initial training of CNN with labeled images only;
while *CNN not converged* **do**
  for *each queried image I* **do**
    select whether to use I for training
  end
  train CNN with labelled and selected images
end

Algorithm 2: Proposed pseudolabel algorithm

5.4.2 Pseudolabels with Query Images

The method described here relies on a different pseudolabel selection mechanism and a different pseudolabel weighting to the original approach [Lee13]. When training with pseudolabel data, the CNN is trained as described in section 5.4.1. However, $\mathbf{X}^{\text{query}}$ images are repeatedly evaluated with the current network, and some are selected with pseudolabels $\mathbf{X}^{\text{pl}}$, for training.

At the beginning of training, $\mathbf{X}_0^{\text{pl}}$ is empty.

$$\mathbf{X}_0^{\text{pl}} = \emptyset \tag{5.2}$$

For the first i iterations (during epoch 0), the CNN is trained only with $(\mathbf{X}^{\text{train}}, \mathbf{y}^{\text{train}})$. Then, $\mathbf{X}_0^{\text{query}}$ is propagated forward through the CNN, to produce a set of vectors of beliefs for all labels $\mathbf{b}_0$ for every query image. These beliefs correspond to the normalized outputs of the last fully connected layer, before applying the last softmax layer.

Then, a randomized selection process chooses which predicted labels $\mathbf{y}^{\text{query}}$ will be trusted. Pseudolabel examples $\mathbf{X}_e^{\text{pl}}$ from the previous epoch are excluded.

$$(\mathbf{X}_{e+1}^{\text{pl}}, \mathbf{y}_{e+1}^{\text{pl}}) = selected(\mathbf{X}^{\text{query}} \setminus \mathbf{X}_e^{\text{pl}}, \mathbf{y}^{\text{query}}, \mathbf{b}_e) \tag{5.3}$$

The selection method proposed here is explained in section 5.4.3. The rest of $\mathbf{X}^{\text{query}} \setminus \mathbf{X}_e^{\text{pl}}$ is unused in this epoch.

This is the end of epoch 0. In each following epoch e, the CNN is trained with $\{(\mathbf{X}_e^{\text{pl}}, \mathbf{y}_e^{\text{pl}}), (\mathbf{X}^{\text{train}}, \mathbf{y}^{\text{train}})\}$. Section 5.4.4 discusses how $\mathbf{y}_e^{\text{pl}}$ can be weighted against $\mathbf{y}^{\text{train}}$ for better convergence stability.

5.4.3 Pseudolabel Selection

Each example image is chosen with probability:

$$\frac{(1-\lambda_c) * b_e}{2} \tag{5.4}$$

Where the accuracy λ_c for each class c on unlabeled data is the ratio of images classified as class c to the number of queried images in class c. By making the weak assumption that queried class accuracies across queried data are similar, class accuracies λ_c for the classifier are an indicator of training data and class complexity for each category.

The classifier belief b_e is the activation of the image for the queried class, as predicted by the network. By using the normalized belief in the $\mathbf{y}^{\text{query}}$ class, the selection favours images the classifier is more confident about, thus removing incorrect query images. This belief is normalized across network responses.

Classes with higher accuracy on the query dataset are given lower pseudolabel priority. This is accomplished with the $(1-\lambda_c)$ factor.

A number of factors affect the quantitative benefit of using pseudolabeled images: dataset belief, the accuracy of the selection method, the difference between datasets, selection variability over epochs, and randomization. This selection method balances these by selecting images in a randomized order, which depends on class accuracies and classifier belief for the correct class.

The last step is randomization. A portion of query images is randomly removed during selection. In these experiments, 50% were removed, and this was found to be beneficial. This is justified by a need to regularize across data when the CNN is trained.

5.4.4 Pseudolabel Weighting

Pseudolabels are likely to affect the classifier adversely when it hasn't yet reached a sufficient accuracy, just as the classifier would fail to train on raw query data. Self-training is prone to quickly converge to suboptimal solutions because the classifier assigns high confidence to wrong examples. How this is mitigated in this approach is explained below.

In the original pseudolabel paper [Lee13], images from the training set have constant weights, and the pseudolabel losses are weighted by α, where α increases with time according to two hyperparameters.

Our experiments showed that this method is not more effective than setting $\alpha = 0$ until the network reaches near-top accuracy and then setting $\alpha = 1$. This method crucially relies on the network's ability to create a weak classifier from the training data alone, and it was found that this is the case with the previously published α tuning method as well. All shown results are achieved with this step function, thus demonstrating its usefulness.

This weighting method, albeit crude, simplifies hyperparameter tuning, and at the cost of a few epochs, achieves the same accuracy.

5.4.5 Dataset Belief

For an automatically retrieved set of images, a crucial piece of information for deciding whether to train using pseudolabels is the accuracy of the queried data. The unknown proportion of images which belong to the queried category is B, or dataset belief.

Query images can be wrong, misleading, and/or contain correctly and incorrectly labelled images from the training dataset, see Figure 5.13.

An imperfect selection must vary over epochs, in order to mitigate convergence to a non-median representation of the category.

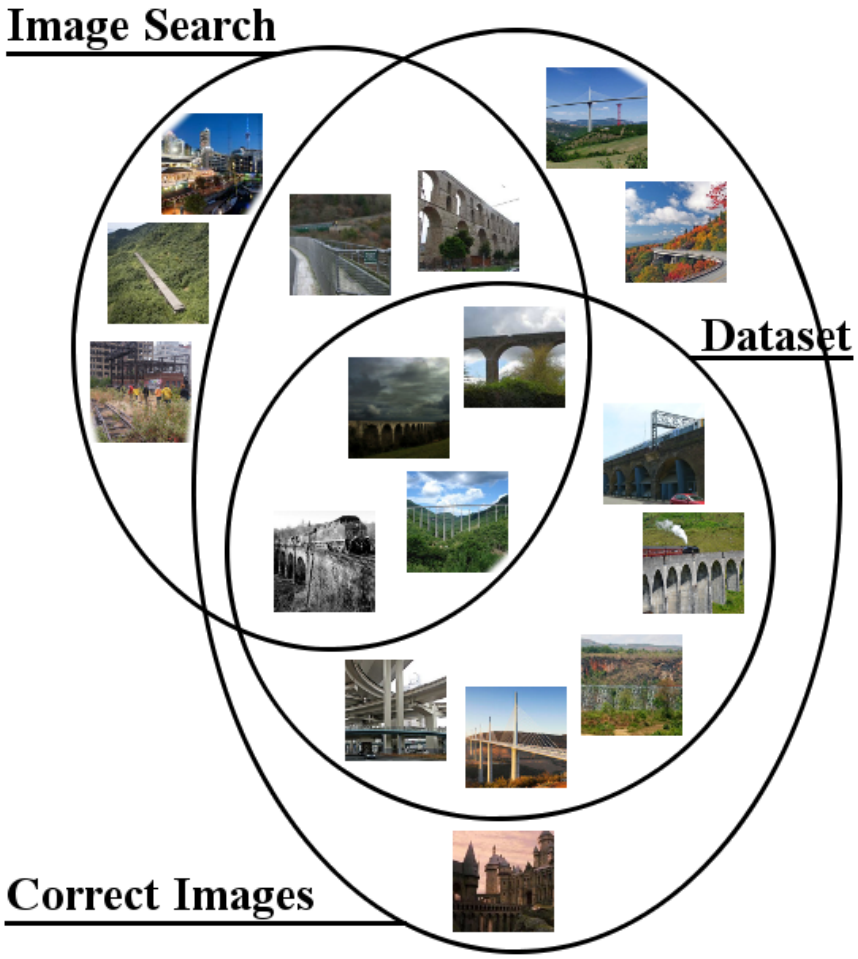


Figure 5.13: Example images of the viaduct class

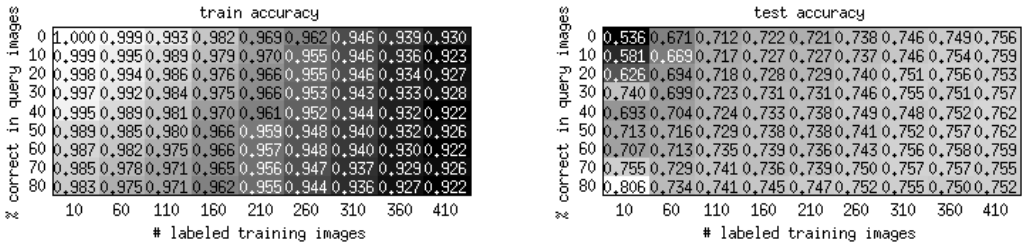


Figure 5.14: Train and test accuracies with varying correct query images, and varying train set sizes for each class

5.4.6 Difference Between Datasets

If the training dataset and the images queried from online image search are the same, the method will not be of benefit. It is important that they are complementary, albeit with an overlap, and that they disagree to a degree. The disagreement creates jitter in the hyperspace between images where the classifier should not be divisive, and it supports convergence to a decision boundary elsewhere.

We found that selecting (X^{query}, y^{query}) which fully agrees with the current classifier does not boost classifier accuracy over not using pseudolabels at all. This is because despite bringing new information, the data doesn't create disagreement, and therefore no novelty. In these experiments, it was found that a certain degree of wrong and randomly labelled images helped the classifier to converge to higher accuracy over the test set. Adding this form of noise achieves regularisation.

5.4.7 Implementation

All images $\mathbf{X}^{\text{train}}, \mathbf{X}^{\text{test}}, \mathbf{X}^{\text{query}}$ were resized so that the smaller dimension is 227 pixels, and a central crop of 227×227 pixels is extracted. This has been shown to work better than other cropping methods [CSVZ14], and the value 227 was chosen because this is the input size of the AlexNet network [KSH12]. Preprocessing details are discussed and evaluated in [CSVZ14].

The AlexNet [KSH12] architecture was used and initialized with weights trained on the ImageNet dataset. The network was retrained by keeping all but the last fully connected layer locked, and by updating weights on the last layer.

The network was trained over 100 epochs of 500 iterations each with each combination of parameters. In a GPU-accelerated environment, such a network on the full SUN dataset with all query images converged in 2 to 5 hours.

The ratio of testing data to queried data accuracies is an indicator of the queried datasets accuracy or similarity. Assuming no constructive errors, such as those CNNs have been demonstrated to fall to when synthesizing examples [NYC15], the number of correctly classified images is a lower bound on how many really belong into the category. A large difference between this number and the actual number (B), directly indicates how much further benefit the new data can have for training.

As shown in the right table 5.14, over test data, it can be seen that when the number of labelled images is small, the Active Learning approach using pseudolabels and a weak classified image retrieval system is of significant benefit. The accuracy can increase by as much as 25%, thus demonstrating that Active Learning benefits the critical Computer Vision task of learning image classifiers. In fact, the broad spectrum of classes and the small amount of data shows that this general approach can benefit many further tasks, well beyond the scope of this experiment.

5.5 Validating the Hypothesis

This chapter lists several experiments in which Active Learning has benefited contemporary Computer Vision, complementing existing algorithms via the judicious application of human labelling effort, or the use of pre-trained models for other, similar tasks.

Specifically, Active Learning has been shown to increase the quality of Generative Adversarial Networks for Font Capture by allowing the preparation of a larger and more representative dataset, it has enhanced the applicability of conditional GANs for generating faces by allowing the control of features, it has reduced the necessary time to manually annotate varied tags on images, and it has been shown to enable weak supervision to vastly improve the classification accuracy of image classifiers.

In terms of the symbolic formulation of the hypothesis, the method has shown that for various problems M and their associated loss functions L, there exists an Active Learning approach S which takes data X to produce parameters for the model which increase its accuracy over L. In practice, S can often benefit M even without the need for significant additional manual annotation, but by efficiently using Transfer Learning of existing algorithms as annotation experts.

The hypothesis, as stated in chapter 4, states that *Human and algorithmic expert annotation using Active Learning improves the accuracy of contemporary Computer Vision methods.* The work presented in this demonstrates this repeatedly for the human expert case, as well as for an algorithmic expert.

Therefore, this work validates the hypothesis formulated in the last chapter, with improvements by a significant margin to several contemporary Computer Vision problems.

CHAPTER 6

Applications and Future Work

Thanks to the methods developed in the last chapter, it will be possible to reduce the annotation time for labelling data for various applications in Computer Vision. There are tasks where this could be done in a fully automated way, by utilizing pre-trained algorithms as the expert oracle, and there are other tasks where the amount of human labelling and tagging can be reduced.

The presented graphical user interface of section 5.3 can serve to annotate context-specific images for improved automated classification. While the approach to iterative optimization of section 5.1 to clean large datasets has been applied before, it has been shown to be of significant benefit and can be applied to other datasets, as will be discussed in section 6.3. I also describe applications which can benefit from the approach presented in section 5.3, where large-scale cross-domain Transfer Learning is systematically applied to train image classifiers.

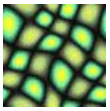
The following sections describe future work in Active Learning. Section 6.1 discusses how Active Learning may improve upon texture synthesis, and section 6.2 describes current work on hyperspectral images.

6.1 Applications in Textures

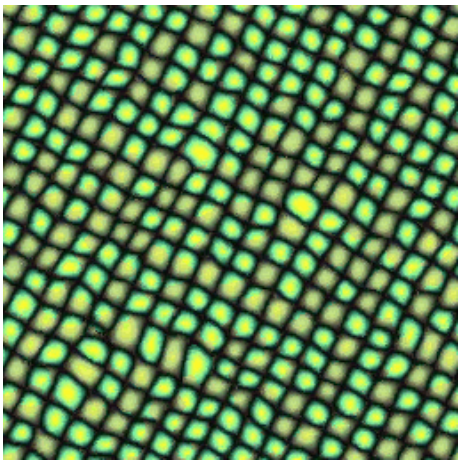
Example-based texture synthesis is an important problem of Computer Graphics, where a small sample image is converted into a non-repeating texture, as shown in figure 6.1.

Numerous algorithms exist for example-based texture synthesis, some focusing of speed, others on memory efficiency, and many on quality. Among algorithms focused on quality, none resolve the issue of tuning quality in locations where it is visually most lacking [KDC17].

This problem may be addressed with Active Learning, allowing annotators to interactively point to visual flaws in generated textures, so that they may be resolved algorithmically. Such an approach would make it possible to improve the quality of texture synthesis methods.



(a) input texture (exemplar)



(b) proceduraly generated texture

Figure 6.1: Example of texture synthesis, taken from own work [KCD15]

6.2 Spectral Interactive Annotation

Hyperspectral data allows the gathering of additional useful information from a scene, allowing for more accurate classification or detection of natural objects or chemical processes. However, additional channels are challenging to visualise intuitively, especially in the case of full-spectrum measurements in every pixel of a scene. Such data is typically processed by analysing spectra of selected pixels, while viewing an RGB visualisation of the entire scene. One such hyperspectral classification platform is HSImager, shown in figure 6.2.

Active Learning may be beneficial in supporting the annotation process, by optimising the RGB visualisation of hyperspectral features. This feature-wise Active Learning approach focuses on presenting individual datapoints, rather than selecting relevant data for annotation. In this sense it differs from the approaches presented throughout this work, where manual annotation focuses on adding features to chosen data. Therefore, the two approaches may also be complementary.

In practice, the annotation of spectral images could be performed as follows: the image may be displayed in RGB for labelling initialisation, then locally processed to highlight relevant regions, and itteratively improved by taking into account new pixel classifications.

Hyperspectral data is increasingly easy to capture, and with a wide range of applications, simplifying annotation while increasing model accuracy is an important research question. Active Learning techniques are an excellent choice for addressing this.

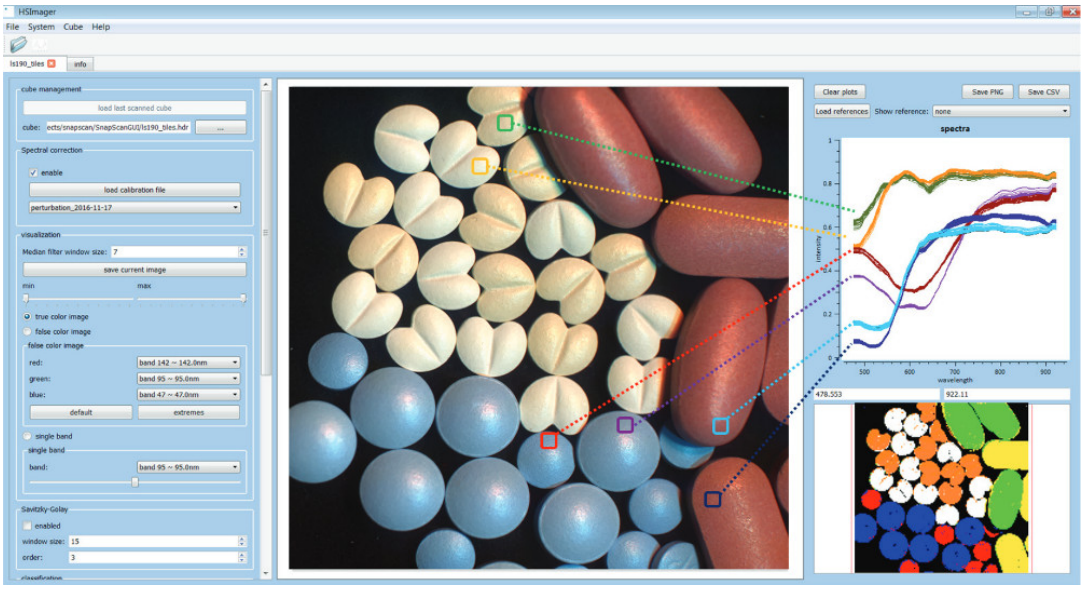


Figure 6.2: Spectral image processing pipeline of HSImager [BM18]. The interface can be improved by implementing a feedback mechanism for which pixels would be beneficial to label.

6.3 Future Work

The individual approaches developed in chapter 5 can be extended to be applied in numerous settings, to improve a wide range of Computer Vision models on various datasets. These are briefly discussed here.

The graphical interface and recommender system of section 5.3 can be used to label a multitude of relevant classes in numerous scenarios. For example, the processing of video feeds can be partially automated to detect people, animals, craft, or traded objects. The system could be used for non-expert labelling of food served in a restaurant, or for the counting of wild animals in a nature reserve.

Whenever a large amount of binary labelled images is needed, the approach presented in section 5.1 to produce a font dataset can be used to semi-automatically produce hundreds of thousands of useful annotations in a few hours. This method suffers from a lack of direct feedback during each iteration, and the expert annotation process could be improved by updating the relevant images after each label is placed, instead of working in large slow batches.

Most generally, the pseudolabel approach of section 5.4 can help train classifiers for all classes described by text, such as all the 117 000 synsets of the WordNet lexical database. More specifically, it can be used to optimize the training of facial recognition, car classification, and any other data where weak labels are easily available on a wide range of images. Just as state-of-the-art models have seen major improvements due to the extension of datasets, a dataset produced with an algorithmic expert could contain a hundred thousand of classes and millions of examples for each, ushering a new era in large-scale Computer Vision.

In the future, introducing Active Learning with human and algorithmic experts into problems of Computer Vision can enable significantly broader and more accurate detections, segmentations, and classifications than before.

CHAPTER 7

Conclusion

Machine Vision model quality is dependent on the versatility of the prior of the models used, on hyperparameters and parameter tuning, and the range and accuracy of data see during training. This work focuses on improving the accuracy of models by increasing data quality and quantity through Active Learning, validates the posed hypothesis, and demonstrates its benefits in a number of scenarios.

These main scientific contribution is the validation of the hypothesis, which stipulates that Active Learning benefits Computer Vision. This hypothesis is validated in two sets of differing scenarios: increasing the efficiency of manual labour for annotation, and utilizing Transfer Learning principles by applying pre-trained models to benefit a task. The applied contribution of this work is a series of experimental demonstrations of the hypothesis, and minor contributions are application-specific model improvements in Font Capture, One-shot-learning for image classification, and a tagging GUI to simplify annotation.

A system for human-assisted Image-wise tagging with suggestions was created, so that it could be used to obtain large semantically labelled datasets. The suggestion methods, as well as the annotating system itself, could be applied in the context of public media databases. The obtained annotations contain a higher percentage of positive examples of infrequent classes.

In another application, font capture benefited from Active Learning. Fonts are present in all forms of visual media, but working with them remains possible only for those with access to the type definitions. This work widens the possibilities for tools such as Photoshop and Google image translate, where recreating text in a given font is key. Automatically expanding the diacritical sets for existing fonts brings all fonts to a wider audience of hundreds of millions of users whose language includes diacritics.

Generative Adversarial Networks for generating faces have also seen an improvement thanks to Active Learning, by which new faces can be rendered with explicitly set features, such as gender and age. This benefit has come thanks to applying knowledge from other pre-trained

models on existing data, showing that Active Learning is also beneficial in the case of algorithmic experts, rather than only with human annotators.

Similarly, an algorithmic expert in image retrieval was integrated into an extended pseudolabel training framework for CNN classifiers, demonstrating that Active Learning will push forward challenging tasks like image classification. This new method also does not require human supervision or annotation, bringing forth the possibility of extended applications by which Active Learning is applied seamlessly in Transfer Learning and Life-long learning tasks.

These specific tasks are some examples where my work has shown the benefit of Active Learning in increasing the quality of contemporary Computer Vision methods. While this validation is not a theoretical solution answering the hypothesis, this work will have demonstrated the general applicability of these principles and will enable a theoretical as well as a practical methodology to increase the quality of Computer Vision models at little cost.

In future work, it may be interesting to explore the question in a fashion systematic enough to allow automatic application, thus allowing the creation of an algorithm which searches for trainable models and existing datasets, and semi-automatically improves them using other known data and models. It will be particularly interesting to apply these principles to the other unsolved tasks where a large knowledge base can be drawn upon, such as theorem-proving, Computer Vision in video, and hyperspectral.

Glossary

Active Learning Process of selecting which data needs to get an expert label, either by a human or by another algorithm. 1, 3–8, 17–19, 25, 32–34, 36, 40, 44, 48, 59

Adaptive Learning Parameters are adjusted at runtime. 44

Association-rule Learning Discovering information about relationships. 33

Class-domain-wise tagging Assigning images to a tag. 20, 31, 44, 47

Collaborative Filtering Predicting preferences of users for objects given sparse preferences by other users. 44

Conformal Predictors Labeling unlabeled data using labels from similar labeled data. 25

Deep Learning Training a sequence of processes to map input to output. 8, 19–21

Federated Learning Distributed training on unshared local data. 9, 10

Hybrid Action Learning Active Learning with unsupervised initialisation. 48

Image-wise tagging Assigning tags to an image. 44, 59

Incremental Learning Test-time input is used to improve the model. 48

Multilabel Learning Data belong to multiple categories simultaneously. 20

Multiple Instance Learning Supervised learning where a labeled set contains at least one instance with that label. 6

Multitask Learning Concurrent transfer learning. 7

One-shot Learning One or few examples while training. 33

Online Learning Training data is not statically available. 48

PU learning Positive and unlabeled data only. 29, 48

Reinforcement Learning Agents learn to take actions. 6

Repetition Priming The sequence in which images are presented to the annotator affects their perception. 20

Semi-supervised Learning A subset of the data is labeled. 1, 6, 7, 30

Supervised Learning Using labeled data. 5–7

Transfer Learning Adapting a pre-trained model. 1, 7, 19, 23, 32, 35, 41, 60

Unsupervised Learning Using unlabeled data. 5

Weakly Supervised Learning on data with noisy, limited, or imprecise labels. 6, 48

Zero-shot Learning Classifying to classes not seen during training. 33

Own work

[HKL+12] Michal Hradiš, Martin Kolář, Aleš Láník, Jiří Král, Pavel Zemčík, and Pavel Smrž. Annotating images with suggestions—user study of a tagging system. In *International Conference on Advanced Concepts for Intelligent Vision Systems*, pages 155–166. Springer, Berlin, Heidelberg, 2012.

[JVK+20] Roman Juránek, Jakub Výravský, Martin Kolář, David Motl, and Pavel Zemčík. Graph-based deep learning segmentation of eds spectral images for automated mineralogy analysis. *Computers and Geosciences, under review*, 2020.

[KCD15] Martin Kolář, Alan Chalmers, and Kurt Debattista. Repeatable texture sampling with interchangeable patches. *The Visual Computer*, 2015.

[KDC17] Martin Kolář, Kurt Debattista, and Alan Chalmers. A subjective evaluation of texture synthesis methods. In *Computer Graphics Forum*, volume 36, pages 189–198, 2017.

[KHZ15] Martin Kolář, Michal Hradiš, and Pavel Zemčík. Technical report: Image captioning with semantically similar images. *arXiv:1506.03995*, 2015.

[KHZ16] Martin Kolář, Michal Hradiš, and Pavel Zemčík. Deep learning on small datasets using online image search. In *Proceedings of the 32nd Spring Conference on Computer Graphics*, pages 87–93, 2016.

[KHZ20] Martin Kolář, Michal Hradiš, and Pavel Zemčík. Capturing fonts in the wild. *The Visual Computer, under review*, 2020.

Bibliography

[ABC+16] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

[AFK+18] Samaneh Azadi, Matthew Fisher, Vladimir G Kim, Zhaowen Wang, Eli Shechtman, and Trevor Darrell. Multi-content gan for few-shot font style transfer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7564–7573, 2018.

[AQ07] Stéphane Ayache and Georges Quénot. Evaluation of active learning strategies for video indexing. *Signal Processing: Image Communication*, 22(7-8):692–704, August 2007.

[AQ08] Stéephane Ayache and Georges Quéenot. Video Corpus Annotation using Active Learning. In *European Conference on Information Retrieval (ECIR)*, pages 187–198, Glasgow, Scotland, March 2008.

[Bal17] Shumeet Baluja. Learning typographic style: from discrimination to synthesis. *Machine Vision and Applications*, 28(5-6):551–568, 2017.

[BEG+19] Keith Bonawitz, Hubert Eichner, W. Grieskamp, Dzmitry Huba, Alex Ingerman, V. Ivanov, Chloé Kiddon, Jakub Konecný, S. Mazzocchi, H. McMahan, Timon Van Overveldt, D. Petrou, D. Ramage, and Jason Roselander. Towards federated learning at scale: System design. *ArXiv*, abs/1902.01046, 2019.

[Ben12] Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade*, pages 437–478. Springer, 2012.

[Ber16] Erik Bernhardsson. Analyzing 50k fonts using deep neural networks. https://erikbern.com/2016/01/21/analyzing-50k-fonts-using-deep-neural-networks.html, 2016.

[BLU+14] Djallel Bouneffouf, Romain Laroche, Tanguy Urvoy, Raphael Féraud, and Robin Allesiardo. Contextual bandit for active learning: Active thompson sampling. In *International Conference on Neural Information Processing*, pages 405–412. Springer, 2014.

[BM18] Ryan C Brown and Joshua Moser. Hsimage: a python and c++ library to allow interaction with envi-bil hyperspectral images. *Journal of Open Source Software*, 3(25):630, 2018.

[Bow81] Adrian Bowyer. Computing dirichlet tessellations. *The computer journal*, 24(2):162–166, 1981.

[BRK19] Samuel Budd, Emma C Robinson, and Bernhard Kainz. A survey on active learning and human-in-the-loop deep learning for medical image analysis. *arXiv*, pages arXiv–1910, 2019.

[BZLM18] Tadas Baltrusaitis, Amir Zadeh, Yao Chong Lim, and Louis-Philippe Morency. Openface 2.0: Facial behavior analysis toolkit. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, pages 59–66. IEEE, 2018.

[CDLFF08] Brendan Collins, Jia Deng, Kai Li, and Li Fei-Fei. Towards scalable dataset construction: An active learning approach. In David Forsyth, Philip Torr, and Andrew Zisserman, editors, *Computer Vision – ECCV 2008*, pages 86–98, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[CFL+15] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollar, and C Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv*, pages arXiv–1504, 2015.

[CGL13] Taylor Childers, Jessica Griscti, and Liberty Leben. 25 systems for classifying typography: A study in naming frequency. *PJIM Parsons Journal for Information Mapping*, 5(1), 2013.

[CK14] Neill DF Campbell and Jan Kautz. Learning a manifold of fonts. *ACM Transactions on Graphics (TOG)*, 33(4):91, 2014.

[CPH05] M A Carreira-Perpinan and G E Hinton. On Contrastive Divergence Learning. In Robert G Cowell and Zoubin Ghahramani, editors, *Artificial Intelligence and Statistics*, page 17. Citeseer, Society for Artificial Intelligence and Statistics, 2005.

[CSVZ14] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *British Machine Vision Conference*, 2014.

[CV95] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[CZ] Olivier Chapelle and Alexander Zien. Semi-supervised classification by low density separation. *AI STATS 2005, Bridgetown, Barbados*.

[CZ14] E. J. Crowley and A. Zisserman. In search of art. In *Workshop on Computer Vision for Art Analysis, ECCV*, 2014.

[DBLFF10] Jia Deng, Alexander C Berg, Kai Li, and Li Fei-Fei. What does classifying more than 10,000 image categories tell us? In *European conference on computer vision*, pages 71–84. Springer, 2010.

[DDS+09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[DJV+14] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655, 2014.

[DWD+16] Shubhomoy Das, Weng-Keen Wong, Thomas Dietterich, Alan Fern, and Andrew Emmott. Incorporating expert feedback into active anomaly discovery. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 853–858. IEEE, 2016.

[Dzz15] Haizhou Du, Shengjie zhao, and Daqiang zhang. Robust local outlier detection. In *Proceedings of the 2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, ICDMW '15, page 116–123, USA, 2015. IEEE Computer Society.

[EEVG+14] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2014.

[FFFP07] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007.

[gan16] A fontastic voyage: Generative fonts with adversarial networks. https://multithreaded.stitchfix.com/blog/2016/02/02/a-fontastic-voyage/, 2016.

[GH07] G Griffin and A Holub. Caltech-256 object category dataset. *Technical Report 7694, California Institute of Technology*, 2007.

[GK72] Wolfgang Gudat and Christof Kunz. Close similarity between photoelectric yield and photoabsorption spectra in the soft-x-ray range. *Physical Review Letters*, 29(3):169, 1972.

[GNM+18] J. Goldstein, Dale Newbury, Joseph Michael, Nicholas Ritchie, John Scott, and David Joy. *Scanning Electron Microscopy and X-ray Microanalysis*. 2018.

[Gol08] Philippe Golle. Machine learning attacks against the asirra captcha. In *Proceedings of the 15th ACM conference on Computer and communications security*, pages 535–542, 2008.

[GPAM+14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[GSC00] Felix A. Gers, Jürgen Schmidhuber, and Fred A. Cummins. Learning to forget: Continual prediction with LSTM. *Neural Comput.*, 12(10):2451–2471, 2000.

[HGS+18] Tomáš Hrstka, Paul Gottlieb, Roman Skála, Karel Breiter, and David Motl. Automated mineralogy and petrology-applications of TESCAN Integrated Mineral Analyzer (TIMA). *Journal of Geosciences*, 63(1):47–63, 2018.

[Hin02] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002.

[Ho95] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.

[HOT06] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

[HRBLM07] Gary B Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. *Technical Report 07-49, University of Massachusetts, Amherst*, 2007.

[ide17] Identifont. http://www.identifont.com, 2017.

[JSD+14] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM International Conference on Multimedia*, MM '14, page 675–678, New York, NY, USA, 2014. Association for Computing Machinery.

[KB15] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

[KBZ+19] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. *arXiv preprint arXiv:1912.11370*, 2019.

[KLA19] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[KLA+20] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8110–8119, 2020.

[KMRW14] Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3581–3589, 2014.

[KMY+16] Jakub Konecný, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *CoRR*, abs/1610.05492, 2016.

[Kok15] Iasonas Kokkinos. Surpassing humans in boundary detection using deep learning. *CoRR*, abs/1511.07386, 2015.

[Kri12] Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto, Technical Report*, May 2012.

[KSF17] Ksenia Konyushkova, Raphael Sznitman, and Pascal Fua. Learning active learning from data. In *Advances in Neural Information Processing Systems*, pages 4225–4235, 2017.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[KVJ87] Michael N Katehakis and Arthur F Veinott Jr. The multi-armed bandit problem: decomposition and computation. *Mathematics of Operations Research*, 12(2):262–268, 1987.

[LB97] Terran Lane and Carla E Brodley. An application of machine learning to anomaly detection. In *Proceedings of the 20th National Information Systems Security Conference*, volume 377, pages 366–380. Baltimore, USA, 1997.

[LBBH98a] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[LBBH98b] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[LBD+89] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[LDTX12] Wen Li, Lixin Duan, Ivor W Tsang, and Dong Xu. Batch mode adaptive multiple instance learning for computer vision tasks. In *CVPR*, pages 2368–2375. IEEE, 2012.

[LDXT11] Wen Li, Lixin Duan, Dong Xu, and Ivor W Tsang. Text-based image retrieval using progressive multi-instance learning. In *ICCV*, pages 2049–2055. IEEE, 2011.

[Lee13] DH Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. *Workshop on Challenges in Representation Learning*, 2013.

[LG13] Xin Li and Yuhong Guo. Adaptive active learning for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

[LGRN09] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th annual international conference on machine learning*, pages 609–616, 2009.

[Li94] Stan Z Li. Markov random field models in computer vision. In *European conference on computer vision*, pages 361–370. Springer, 1994.

[LMB+14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014*, pages 740–755. Springer, 2014.

[LVKB19] Alex Lamb, Vikas Verma, Juho Kannala, and Yoshua Bengio. Interpolated adversarial training: Achieving robust neural networks without sacrificing too much accuracy. In *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, AISec'19, page 95–103, New York, NY, USA, 2019. Association for Computing Machinery.

[LWPT11] James L Little, Antony J Williams, Alexey Pshenichnov, and Valery Tkachenko. Identification of "known unknowns" utilizing accurate mass data and chemspider. *Journal of the American Society for Mass Spectrometry*, 23(1):179–185, 2011.

[MF13] David Motl and Vojtěch Filip. Method and apparatus for material analysis by a focused electron beam using characteristic X-rays and back-scattered electrons, February 28 2013. US Patent App. 13/398,114.

[Mil95] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[MMK+16] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, Ken Nakae, and Shin Ishii. Distributional smoothing with virtual adversarial training. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

[MMR+16] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data, 2016.

[MOT15] Alexander Mordvintsev, Christopher Olah, and Mike Tyka. Inceptionism: Going deeper into neural networks. https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html, 2015.

[MSA+11] Jean-Baptiste Michel, Yuan Kui Shen, Aviva Presser Aiden, Adrian Veres, Matthew K Gray, Joseph P Pickett, Dale Hoiberg, Dan Clancy, Peter Norvig, Jon Orwant, et al. Quantitative analysis of culture using millions of digitized books. *science*, 331(6014):176–182, 2011.

[MSDC12] Lázaro Emílio Makili, Jesús A Vega Sánchez, and Sebastián Dormido-Canto. Active learning using conformal predictors: application to image classification. *Fusion Science and Technology*, 62(2):347–355, 2012.

[Mun20] Robert Munro. In Grégoire Montavon, Genevieve B Orr, and Klaus-Robert Müller, editors, *Human-in-the-Loop Machine Learning*, MEAP. Manning, 2020.

[NH10] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 807–814, Madison, WI, USA, 2010. Omnipress.

[NYC15] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. *CVPR*, 2015.

[OAM+11] Paul Over, George Awad, Martial Michel, Jonathan Fiscus, Wessel Kraaij, Alan F. Smeaton, and Georges Quéenot. TRECVID 2011 – An Overview of the Goals, Tasks, Data, Evaluation Mechanisms and Metrics. In *Proceedings of TRECVID 2011*. NIST, USA, 2011.

[OBLS14] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Weakly supervised object recognition with convolutional neural networks. Technical Report HAL-01015140, INRIA, 2014.

[OLAH14] Peter O'Donovan, Jānis Lībeks, Aseem Agarwala, and Aaron Hertzmann. Exploratory font selection using crowdsourced attributes. *ACM Transactions on Graphics (TOG)*, 33(4):92, 2014.

[ONJ13] Salima Omar, Asri Ngadi, and Hamid H Jebur. Machine learning techniques for anomaly detection: an overview. *International Journal of Computer Applications*, 79(2), 2013.

[PCD08] Nicolas Pinto, David D Cox, and James J DiCarlo. Why is real-world visual object recognition hard? *PLoS Comput Biol*, 4(1):e27, 2008.

[PVZ15] Omkar M. Parkhi, Andrea Vedaldi, and Andrew Zisserman. Deep face recognition. In Xianghua Xie, Mark W. Jones, and Gary K. L. Tam, editors, *Proceedings of the British Machine Vision Conference 2015, BMVC 2015, Swansea, UK, September 7-10, 2015*, pages 41.1–41.12. BMVA Press, 2015.

[RDS+15a] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 2015.

[RDS+15b] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 2015.

[Rit09] Nicholas W.M. Ritchie. Spectrum simulation in DTSA-II. *Microscopy and Microanalysis*, 15(5):454–468, 2009.

[RMC16] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

[Roo17] Kris Roosen. Gan in keras on mnist. https://github.com/kroosen/GAN-in-keras-on-mnist, 2017.

[SCF08] Burr Settles, Mark Craven, and Lewis Friedland. Active learning with real annotation costs. In *Proceedings of the NIPS workshop on cost-sensitive learning*, pages 1–10. Vancouver, CA:, 2008.

[SED19] Samarth Sinha, Sayna Ebrahimi, and Trevor Darrell. Variational adversarial active learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5972–5981, 2019.

[SEZ+13] Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, and Yann LeCun. OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks. December 2013.

[SGZ+16] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242, 2016.

[SKFH16] Jost Tobias Springenberg, Aaron Klein, Stefan Falkner, and Frank Hutter. Bayesian optimization with robust bayesian neural networks. In *Advances in neural information processing systems*, pages 4134–4142, 2016.

[SKP15] Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A unified embedding for face recognition and clustering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823. IEEE, Jun 2015.

[SLJ+14] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going Deeper with Convolutions. *CoRR*, abs/1409.4, 2014.

[SMH07] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. Restricted Boltzmann machines for collaborative filtering. *Proceedings of the 24th International Conference on Machine Learning (2007)*, pp:791–798, 2007.

[Smo86] Paul Smolensky. Information processing in dynamical systems: Foundations of harmony theory. Technical report, Colorado Univ at Boulder Dept of Computer Science, 1986.

[SMV+19] Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller. *Explainable AI: interpreting, explaining and visualizing deep learning*, volume 11700. Springer Nature, 2019.

[squ17] Font identifier. https://www.fontsquirrel.com/matcherator, 2017.

[SUB96] Marcos Salganicoff, Lyle H Ungar, and Ruzena Bajcsy. Active learning for vision-based robot grasping. *Machine Learning*, 23(2-3):251–278, 1996.

[SZ14] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[SZS+13] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6, 2013.

[TE11] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *CVPR*, pages 1521–1528. IEEE, 2011.

[TF97] Joshua B Tenenbaum and William T Freeman. Separating style and content. In *Advances in neural information processing systems*, pages 662–668, 1997.

[TFF08] Antonio Torralba, Rob Fergus, and William T Freeman. 80 Million Tiny Images: a Large Data Set for Nonparametric Object and Scene Recognition. *IEEE transactions on pattern analysis and machine intelligence*, 30(11):1958–70, November 2008.

[The11] The Unicode Consortium. The Unicode Standard. Technical Report Version 10.0.0, Unicode Consortium, 2011.

[TVDJ20] Hugo Touvron, Andrea Vedaldi, Matthijs Douze, and Hervé Jégou. Fixing the train-test resolution discrepancy: Fixefficientnet. *arXiv*, pages arXiv–2003, 2020.

[TYRW14] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *CVPR*, pages 1701–1708. IEEE, 2014.

[USB16] Paul Upchurch, Noah Snavely, and Kavita Bala. From a to z: Supervised transfer of style and content using deep neural network generators. *arXiv*, pages arXiv–1603, 2016.

[VDSGS10] Koen EA Van De Sande, Theo Gevers, and Cees GM Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1582–1596, 2010.

[Ven20] Tomáš Venkrbec. Generating faces with conditional generative adversarial networks. *VUT FIT Bachelor Thesis*, 2020.

[VG08] Sudheendra Vijayanarasimhan and Kristen Grauman. Keywords to visual categories: Multiple-instance learning forweakly supervised object categorization. In *CVPR*, pages 1–8. IEEE, 2008.

[VSK+19] Neha Verma, Vishal Sharma, Raj Kumar, R Sharma, MC Joshi, GR Umapathy, Sunil Ohja, and Sundeep Chopra. On the spectroscopic examination of printed documents by using a field emission scanning electron microscope with energy-dispersive x-ray spectroscopy (fe-sem-eds) and chemometric methods: application in forensic science. *Analytical and bioanalytical chemistry*, 411(16):3477–3495, 2019.

[wik18] List of languages by number of native speakers. https://en.wikipedia.org/wiki/List_of_languages_by_number_of_native_speakers, 2018.

[WWM+06] Christopher G Worley, Sara S Wiltshire, Thomasin C Miller, George J Havrilla, and Vahid Majidi. Detection of visible and latent fingerprints using micro-x-ray fluorescence elemental imaging. *Journal of forensic sciences*, 51(1):57–63, 2006.

[WYS+15] Ren Wu, Shengen Yan, Yi Shan, Qingqing Dang, and Gang Sun. Deep image: Scaling up image recognition. *arXiv*, pages arXiv–1501, 2015.

[WZZ+13] Li Wan, Matthew Zeiler, Sixin Zhang, Yann L Cun, and Rob Fergus. Regularization of neural networks using dropconnect. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 1058–1066, 2013.

[XHE+10] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, pages 3485–3492. IEEE, 2010.

[YCBL14] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems*, pages 3320–3328, 2014.

[YCN+15] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *arXiv*, pages arXiv–1506, 2015.

[YK19] Donggeun Yoo and In So Kweon. Learning loss for active learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 93–102, 2019.

[YRLT] Jenny Yuen, Bryan Russell, Ce Liu, and Antonio Torralba. LabelMe video : Building a Video Database with Human Annotations. *Event (London)*, pages 1–8.

[YSZ+15] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop, 2015.

[ZLL+18] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

[ZLLS19] Aston Zhang, Zachary C Lipton, Mu Li, and Alexander J Smola. Dive into deep learning. *Unpublished Draft. Retrieved*, 19:2019, 2019.

[ZZY17] Zhedong Zheng, Liang Zheng, and Yi Yang. Unlabeled samples generated by gan improve the person re-identification baseline in vitro. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3754–3762, 2017.