DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

Imputation of missing Product Information using Deep Learning: A Use Case on the Amazon Product Catalogue

Aamna Najmi

DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

Imputation of missing Product Information using Deep Learning: A Use Case on the Amazon Product Catalogue

Imputation fehlender Produktinformationen mithilfe von Deep Learning: Ein Anwendungsfall im Amazon-Produktkatalog

Author:	Aamna Najmi
Supervisor:	Prof. Dr. rer. nat. Florian Matthes
Advisor:	Ahmed Elnaggar
Submission Date:	15.06.2019

I confirm that this master's thesis in informatics is my own work and I have documented all sources and material used.

Munich, 15.06.2019 Aamna Najmi

Acknowledgments

I would like to thank Prof. Dr. rer. nat. Florian Matthes for granting me the opportunity to work on this topic at his chair Software Engineering for Business Information Systems (sebis). I would also like to thank my advisor Ahmed Elnaggar for his continuous feedback and motivation throughout my thesis. I am fortunate to get useful insights and perspective as well as continuous feedback from him.
I would like to extend my gratitude to all the faculty members of this institution and my colleagues for their continuous guidance and support during my tenure as a student.
I would also like to take this opportunity to thank my family, especially my mother, father and my partner for believing in me and supporting me in all my endeavours. Lastly, I would like to thank my friends for being there through difficult times and sharing the good ones.

Abstract

The last couple of years have seen massive advancement in deep learning across many tasks such as computer vision (CV), natural language processing (NLP) and speech recognition. This advancement can be observed by end users across various online platforms, one such platform being the e-commerce domain where giants like Amazon are providing users voice assistants, personalized recommendations and efficient product search options. The advancement in the field of deep learning has been catalyzed by the availability of enormous annotated datasets like the Wikipedia corpora in various languages [1] and the Imagenet dataset [2]. However, there have not been appropriate amount of pre-processed datasets in the e-commerce domain that are available for research. With the customer being the most significant part on any e-commerce platform, there is a rising need of natural language and computer vision enabled applications to improve user experience and increase organizational benefits.

In order to overcome the shortage of publicly available datasets in the e-commerce domain in both textual and visual form, we propose the use of domain adaptation to leverage existing advancements in approaches like transfer learning and multi-task learning by using state of the art techniques. Domain adaptation exploits task-independent commonalities and overcomes the problem of dataset shortage [3], especially in the e-commerce domain. Through this work, we have tried to improve product catalog quality by predicting missing product information such as category, color, brand and target gender on the e-commerce platform thus enabling efficient product search and improving user experience on the platform.

As part of the dataset generation phase, we have created three different e-commerce dataset in languages including English, German and French for text based problems and English, German and Italian for image based problems. The dataset has been used to predict missing product information using deep learning approaches like transfer learning and multi-task learning. We have also compared single task approaches for image classification tasks with transfer learning and discussed benefits. In the natural language processing front, we have compared single task learning with both transfer learning and multi-task learning. We observed that for image classification tasks, single task is on equal footing with transfer learning however the latter is trained and implemented in less than half the time invested in training a deep learning model from scratch. For text classification the text corpora was trained on a state-of-the-art deep learning model, the Transformer. In addition, we compared two types of domain adaptation techniques, transfer learning and multi-task learning and found that both approaches are on an equal footing in terms of accuracy. We show that multi-task and transfer learning is advisable in situations where training data is sparse through experiments in which a jointly trained transformer is able to outperform a single-task trained transformer.

After the predictions, we conducted a survey to see if including the predicted features in the product detail pages helps online customers in making buying decisions. Majority of the respondents prefer the predicted features to be included on the product detail page. Hence, suggesting that the predictions made through transfer learning and multi-task learning are useful and applicable in the e-commerce domain to enhance user experience.

Through this thesis, we show how domain adaptation techniques outperform single task learning for text based datasets in terms of accuracy and f1-score and converges way faster for image classification tasks using the e-commerce datasets. These techniques are better options when dealing with dataset shortage, imbalanced classes and in cases where we do not want to train a model from scratch for a prolonged period of time.

Contents

Acknowledgments **iii**

Abstract **iv**

1. Introduction **2**
1.1. Overview . . . 2
1.2. Motivation . . . 2
1.3. Problem Statement . . . 3
1.4. Research Question . . . 4
1.5. Methodology . . . 5
1.6. Timeline . . . 6
1.7. Structure of the Thesis Document . . . 7

2. Foundations **8**
2.1. Background of Deep Learning . . . 8
2.1.1. Neural Networks: Building Blocks . . . 8
2.1.2. Neural Networks: Methodologies . . . 11
2.2. Natural Language Processing . . . 11
2.3. Computer Vision . . . 13
2.4. Neural Networks: Approaches . . . 15
2.4.1. Single Task Learning . . . 15
2.4.2. Domain Adaptation . . . 15

3. Related Work **20**
3.0.1. Transfer Learning . . . 20
3.0.2. Multi-task learning . . . 24

4. Dataset **28**
4.1. Benchmark Dataset . . . 28
4.2. Scraping the E-commerce platforms . . . 28
4.3. Preprocessing . . . 31
4.3.1. Text based Datasets . . . 31
4.3.2. Image based Datasets . . . 32
4.4. Statistics: Text based Datasets . . . 32
4.4.1. Overview . . . 33
4.4.2. Input Length . . . 33
4.4.3. Target Labels . . . 37

4.5. Statistics: Image based Datasets . . . 40

5. Approaches **44**
5.1. Single Task Learning . . . 44
5.1.1. Text based datasets . . . 44
5.1.2. Image based datasets . . . 47
5.2. Transfer Learning . . . 49
5.2.1. Text based datasets . . . 49
5.2.2. Image based datasets . . . 51
5.3. Multi-task Learning . . . 52
5.3.1. Text based datasets . . . 52

6. Experimental Setup and Results **54**
6.1. Experimental Setup . . . 54
6.1.1. Hardware . . . 54
6.1.2. Hyperparameters . . . 55
6.1.3. Software . . . 55
6.1.4. Evaluation Metrics . . . 56
6.2. Experimental Results . . . 57
6.2.1. Single Task Learning . . . 58
6.2.2. Transfer Learning . . . 60
6.2.3. Multi-task Learning . . . 63

7. Discussion **64**
7.1. Overview . . . 64
7.1.1. Across Task Languages . . . 65
7.1.2. Across Task Types . . . 65
7.1.3. Across Task Modalities . . . 66
7.1.4. Across Task Approaches . . . 66
7.2. Survey . . . 69

8. Conclusion **70**

9. Future Research **72**

A. Appendix **73**
A.1. Figures . . . 73

List of Figures **76**

List of Tables **78**

Bibliography **79**

Abbreviations

GPU	Graphical Processing Unit
NLP	Natural Language Processing
CV	Computer Vision
STL	Single Task Learning
MTL	Multi-task Learning
TL	Transfer Learning
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short Term Memory Network
RMSE	Root Mean Squared Error
BERT	Bidirectional Encoder Representations from Transformers
ReLU	Rectified Linear Unit
ULM-fit	Universal Language Model Fine-tuning for Text Classification
MLP	Multi Layer Perceptron
SGD	Stochastic Gradient Descent
POS	Part of Speech
NER	Named Entity Recognition
SVM	Support Vector Machine
KNN	K Nearest Neighbour Algorithm
LM	Language Model

1. Introduction

1.1. Overview

In the last few decades, e-commerce has grown massively. Global retail e-commerce sales is said to reach about $4 trillion by 2020, accounting to 14.6% of the total spending worldwide [4]. The advancement of artificial intelligence in the last 20 years or so has enabled organizations to sense, predict, develop and automate processes and new technologies. Artificial Intelligence today enables e-commerce websites to recommend personalized products and search for them using conversational language and images as if talking to a real person. This personalized experience of customers has led to the evolution of e-commerce today and resulted in the need of computers to understand complex data like natural language and images. A lot of research has been done in the recent past in the field of natural language understanding and computer vision. With the massive growth of e-commerce, it is indeed interesting to apply state-of-the-art approaches in the field of natural language processing and computer vision in the e-commerce domain. This shall enhance customer experience and increase organizational benefits.

1.2. Motivation

Artificial Intelligence grew massively after surviving a stagnation known as the AI winter in the 1970s with the rise of 'expert systems' in the 1980s followed by Deep Blue, when the first computer chess-playing system defeated a reigning world chess champion, Garry Kasparov [5]. With the start of the 21st century, larger amounts of data known as 'big data' and faster computers enabled Machine Learning to be applied in various sectors. McKinsey Global Institute estimated in their famous paper *"Big data: The next frontier for innovation, competition, and productivity"*, "by 2009, nearly all sectors in the US economy had at least an average of 200 terabytes of stored data" [6]. By 2016, AI related products including hardware and software reached 8 billion dollars worldwide and New York Times reported that the interest in Artificial Intelligence had reached a *'frenzy'* [6].

A branch of Machine Learning called Deep Learning emerged and gained popularity during this time due to its application in various fields including computer vision, natural language processing, medical applications, robotics and speech recognition. Deep Learning involves the application of artificial neural networks (ANN) that are stacked together to form a hierarchical structure of interconnected components that learn from huge datasets over time. Though Deep Learning gained popularity very recently, ANNs have been in the pictures for decades. The first ANN was developed in 1962 and later the idea of backpropagation

algorithm was introduced in 1986. Deeper and more complex architectures like Convolution Neural Networks (CNN) and Recurrent Neural Networks (RNN) came into existence in the 21st century. These architectures could provide solutions for image and text related tasks that were comparable to humans and even better at times. One of the key challenges in Deep Learning applications was hardware support available at the time. A major breakthrough was the introduction of Graphical Processing Units (GPU) by NVIDIA in 2009 that reduced implementation time from weeks to days and paved the path for more efficient and optimized algorithms using specialized hardware and approaches. This has led to enabling the renewal of interest in the field of artificial intelligence especially in deep learning in the last couple of years. The application of this field has grown leaps and bounds ever since, with organizations all over the world employing deep learning in image and text related tasks. The applications of these techniques are visible in the e-commerce domain as well with giants like Amazon introducing personalized recommendations on the platform and launching speech enabled devices that assist in intuitive tasks. The availability of huge amounts of data through e-commerce domains and the rising demands of e-commerce users, makes e-commerce a definitive fit for the application of Deep Learning and Artificial Intelligence as a whole. Breakthrough approaches like multi-task learning and transfer learning can provide huge benefits when dealing with challenges like limited annotated data in the e-commerce domain. Applying solutions implemented on source domain to target domain can result in massive performance gain. This is one of the key motivating factors behind this thesis. Deep Learning is currently at its boom and there is even more need and scope for growth and improvement. Our motivation lies in exploring current solutions and applying them to resolve existing problems of poor product catalog quality and increase traffic on the platform by providing satisfactory user experience.

1.3. Problem Statement

E-commerce has grown massively over the recent years with retail e-commerce sales worldwide estimated to reach 4.88 trillion dollars by 2020 constituting to 14.6% of total retail spending [4]. With the growth in the e-commerce industry, various e-commerce organizations are improving the e-commerce experience by employing technical advancements to enhance the end-user experience. The main focus of an e-commerce website is to attract customers and guide them to the right products. A lot of times, customers face problems in reaching to the desired product due to the low quality of the product catalog. natural language processing techniques and computer vision can be used in such scenarios to predict missing product information which can be useful to the customer in making buying decisions. The fashion department is affected the most by this as there is a diverse range of products available and each product is unique in its own attributes; be it the color, the style, the brand or the target gender the product is aimed at. If a product does not have these vital information displayed on the website, a customer would not want to buy it. The most important part of the product information on an e-commerce site is also the title of the product. The title of the product should be attractive yet precise enough for the customer to know if it is exactly

what she wants. The faster she can get to the right product, the better the chance of her to buy it and come back to the website in the future. The vital attributes of the products can be predicted using the textual information displayed on the website or the image of the product through natural language processing and computer vision respectively. However, there are many challenges to this problem. Computer vision and natural language processing tasks employ Deep Learning methods. Deep Learning requires vast amount of training data in order to achieve desirable results in terms of performance. Currently, there are very limited e-commerce product datasets that are massive and publicly available for research. The scarcity of datasets both in the form of text and image is a big challenge for application and further developments of Deep Learning in the e-commerce domain. Moreover, the challenge lies in producing annotated datasets that can be used for the training of supervised tasks. The scarcity of massive annotated datasets is countered by employing specialized methods in Deep Learning like domain adaptation. We try to solve this problem by using transfer learning and multi-task learning techniques to overcome limited data problem as well as unbalanced dataset problem. Deep Learning requires state of the art computational tools and hardware as it involves a few million or even hundreds of millions of parameters that need to be initialized and updated. For effective and successful training, advanced hardware involving GPUs are required. A huge variety of advanced computational interfaces and tools are now available for use in the field of Deep Learning. Through this thesis, we will apply state of the art parallel training methods and GPU hardware for solving the tasks at hand.

1.4. Research Question

In the recent past, massive improvements in performance across many tasks have been made through deep learning techniques like transfer learning and multi-task learning. The focus of the thesis will be to compare training using multiple tasks concurrently and transfer learning with training on single tasks and to see if the former boosts performance and overcomes problems like imbalanced labels, limited labeled data, etc. Previous studies and research have shown that there are a lot of techniques and approaches that can be employed while using transfer learning and multi-task learning respectively. With the help of various literature reviews, ideal architecture and hyperparameter choices have been chosen for each of the techniques. More insights on which were chosen and why is provided in the later chapters.

After selecting state of the art techniques with best combination of architecture and hyperparameter choices, different experiments were conducted and evaluated in order to extract if there are benefits from using transfer learning and multi-task learning compared to single task learning. Primarily, the experiments conducted tried to gauge if multi-task learning and transfer learning can perform better compared to its counterparts and be useful in the e-commerce domain. Necessary steps for this objective are expressed through the research questions below.

1. Could multi-task learning and transfer learning perform better than single task learning on the Amazon Product Catalog dataset?

2. What architecture choices and hyperparameters shall we use in both multi-task learning and transfer learning to obtain good performance?

3. Can transfer learning and multi-task learning be useful in the e-commerce domain to enhance user-experience?

1.5. Methodology

The various stages of this thesis are summarized below:

- Generation of the e-commerce dataset: At this stage, we prepared the e-commerce dataset by scraping four regional Amazon websites namely France (FR), Germany (DE), Italy (IT) and the United Kingdom (UK) to get product description and image of products belonging to certain categories in the fashion department. We scraped about 100k-200k data points from each of the regional website and parsed it to get information like product information, category, color, brand and item specifications. For the text based tasks, we use the DE, FR and UK datasets whereas for the image based tasks, we use the DE, UK and IT datasets.

- Preparing and integrating the dataset: The dataset pipeline was prepared at this stage. The tasks were divided into two broad categories namely image and text. For the image tasks, the images were divided into respective label folders and the folders were divided into training and validation folders for the training and validation stage. For the text tasks, the text inputs like title, product description, bullet points were concatenated and tokenized at word level. The labels for each text task were integer encoded and the input-label pairs were integrated for respective training tasks.

- Understand and modify state of the art architectures: The existing state of the art architectures for single task learning, transfer learning and multi-task learning were deeply studied and modified in order to incorporate customized datasets and train models on them. We were able to successfully integrate e-commerce dataset for both text and image related tasks and tried ideal combinations of architectures and models which we studied from different related literary work and experiments to see the effects on performance.

- Train models on single task, transfer learning and multi-task learning architectures: After understanding and modifying the state of the art architectures for the generated datasets in hand, we implemented the training phase for single task, transfer learning and multi-task learning to assess their performance. For transfer learning, we used pre-trained weights, fine tuned the model and compared the results with models that were trained from scratch and did not employ any kind of transfer learning approach. For multi-task learning, we trained twelve text related tasks in three languages namely, French, German and English at once along with the language modeling task for the Wikipedia corpora for German, French, Romanian and English language. The results

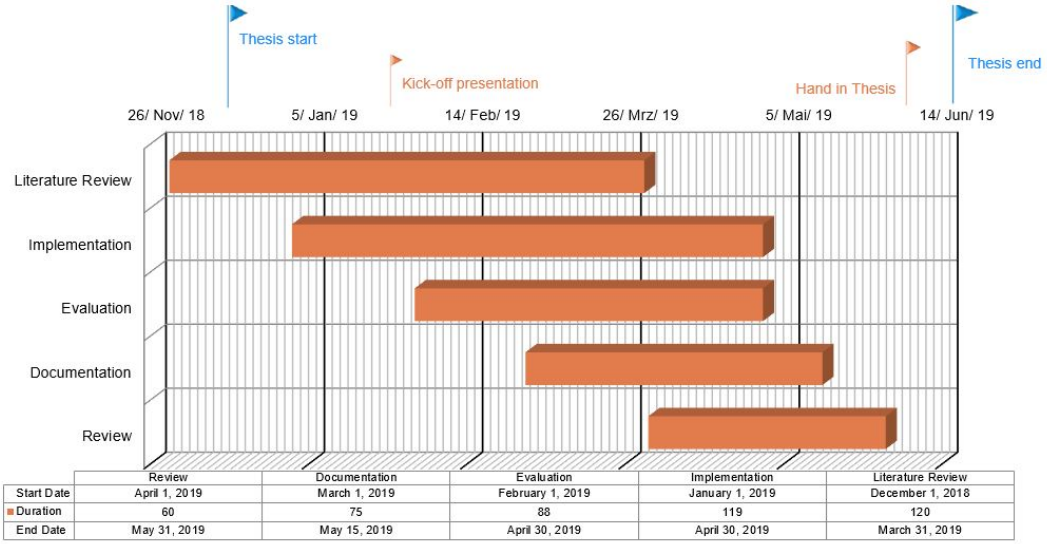
were compared with the results from training each of the twelve tasks independently using the same model architecture. Moreover, transfer learning and multi-task learning were also compared for these tasks of predicting missing product labels.

- Evaluate the results and compare them to their counterparts: The results from each type of task, i.e. single task, multi-task learning and transfer learning for text based and image based tasks were compared using accuracy and f1-scores as the evaluation metrics. The results obtained through the experiments can be found in chapter 6.
- Verify and validate the research questions: After conducting the experiments, the initial research questions were answered and the initial hypotheses were validated. The results were compared and the discussions for the comparison across different languages, modalities, task types and approaches can be found in chapter 7.

Figure 1.1.: A figure depicting the stages of the thesis work.

1.6. Timeline

This section provides an incremental view of the milestones reached during the progress of this project. Figure 1.2 depicts the major milestones. The total time period spent on the thesis work was about six months. The project was started with detailed literature review and analysis to research on current developments and findings in the field of natural language processing and computer vision using domain adaptation techniques like transfer learning and multi-task learning. This was followed by the generation of the dataset from scratch by scraping online e-commerce platforms for about a month. After the generation of the dataset, followed the implementation phase where single task, transfer learning and multi-task learning were applied by employing the dataset in hand. The models were evaluated and based on the evaluations and findings, improvements were applied to achieve gain in performance of the trained models. The final models were reviewed and compared in order to arrive to the best approach in terms of accuracy and f1-score which are important metrics used in classification tasks.



	Review	Documentation	Evaluation	Implementation	Literature Review
Start Date	April 1, 2019	March 1, 2019	February 1, 2019	January 1, 2019	December 1, 2018
Duration	60	75	88	119	120
End Date	May 31, 2019	May 15, 2019	April 30, 2019	April 30, 2019	March 31, 2019

Figure 1.2.: A gantt chart showing the timeline of the thesis.

1.7. Structure of the Thesis Document

This section briefly discusses how the thesis document is structured across different chapters. After the introduction provided above, chapter 2 explains the basic foundations needed to pursue the thesis work. In particular, it breaks down the building blocks of neural networks and the methodologies involved in deep learning architectures. This is followed by discussing natural language processing, computer vision and domain adaptation techniques like transfer learning and multi-task learning. The next chapter, chapter 3, discusses the works done in the field of transfer learning and multi-task learning that has helped us implement our work. Chapter 4 introduces the datasets that have been prepared from scratch for the thesis work along with some statistical information related to the datasets. We then go on to discuss the three approaches that we shall implement through our thesis work followed by the experimental setup behind the experiments and the results achieved in chapter 5 and chapter 6. Chapter 7 discusses the results and compares it with the different approaches implemented. The thesis is concluded with a brief discussion outlining the work done and results achieved followed by future scope of the thesis in chapter 8 and chapter 9.

2. Foundations

In this chapter, we shall discuss the foundations of Deep Learning and its application in the fields of natural language processing and computer vision. The advancements in natural language processing and computer vision has been massive in the recent years where a number of researchers have achieved brilliant performance results on various tasks on a number of publicly available datasets.

2.1. Background of Deep Learning

Machine Learning has seen tremendous amount of growth and development in the industry as well as in research over the years. Usually, machine learning involves a task that is performed by developing a model trained on a set of data points and its efficiency is measured against a validation set. This process is repeated iteratively until the threshold performance is reached. Machine Learning can be broadly categorized into supervised, unsupervised and reinforcement learning. On one hand in supervised learning, the model learns to map the labeled data points to a class label, on the other hand, in unsupervised learning the model learns underlying structure of unlabeled data [7]. Another category of machine learning is reinforcement learning where an agent interacts with the environment and learns actions to maximize the rewards it gets from the environment. Deep Learning also known as hierarchical learning, is a sub-field of Machine Learning that enables machines to learn from experience and mimic human intelligence. Unlike task-specific algorithms, Deep Learning is based on learning underlying data representations [8]. The word 'Deep' is the very core of this approach as numerous layers are stacked together to form deep layers that extract features from data points. The initial layers extract simple concepts while the deeper ones extract more complex concepts that enable these powerful architectures to perform tasks comparable to human performance [9]. In the subsections below, an overview of different building blocks of neural networks is given.

2.1.1. Neural Networks: Building Blocks

- Multi Layer Perceptron: The most popular and highly used deep learning network is the Multi Layer Perceptron (MLP) also known as the Feed Forward Network. A perceptron is a linear classifier that classifies an input by separating two categories with the help of a straight line. Input is typically a feature vector x multiplied by weights w and added to a bias b. A perceptron produces a single output based on several real-valued inputs by forming a linear combination using its input weights (and sometimes passing the

output through a nonlinear activation function). The perceptron formula can be seen below:

$$y = \varphi(\sum_{i=1}^{n} w_i x_i + b) = \varphi(\mathbf{w}^T \mathbf{x} + b)$$

where w denotes the vector of weights, x is the vector of inputs, b is the bias and ϕ is the non-linear activation function [10]. A Multi layer Perceptron is a deep network that consists of a combination of perceptrons. It consists of an input layer through which the input vector is fed and an output layer which makes the predictions. In between these two layers, lie a number of hidden layers that form the computational core of the neural network. Multilayer perceptrons are often applied to supervised learning problems: they train on a set of input-output pairs and learn to model the correlation (or dependencies) between those inputs and outputs. Training involves adjusting the parameters, or the weights and biases, of the model in order to minimize error. Backpropagation is used to make those weight and bias adjustments relative to the error, and the error itself can be measured in a variety of ways, including by root mean squared error (RMSE) [11]. In the forward pass, the signal flow moves from the input layer through the hidden layers to the output layer, and the decision of the output layer is measured against the ground truth labels. In the backward pass, using backpropagation and the chain rule of calculus, partial derivatives of the error function w.r.t. the various weights and biases are back-propagated through the MLP. That act of differentiation gives us a gradient, or a landscape of error, along which the parameters may be adjusted as they move the MLP one step closer to the error minimum. This can be done with any gradient-based optimisation algorithm such as stochastic gradient descent (SGD). This is continued till a point that error cannot get a lower value also commonly known as convergence.

- Convolution Neural Network: Convolution Neural Networks (CNNs), are a type of deep neural networks that are based on a mathematical concept called convolution which helps the structure to take advantage of two dimensional input data points. Convolution Neural Networks are used widely in image related tasks where filters are applied to images to produce feature or activation maps. These convolutions help extract spatial information from images hence making it apt for computer vision. CNNs contain one or more such convolutional layers followed by a fully connected layer in the end that finally makes the predictions [12]. The dimension of the receptive field of the network is much smaller than the input layer and hence there are way fewer connections between inputs and outputs as compared to a fully connected feed forward network. In addition to the convolution operation, there also exists non-linear functions and pooling functions that also help CNNs to take advantage of the two dimensional structure of the data. This also makes the network insensitive to small variations in the input images like rotation or flipping [13]. A number of convolutional layers are stacked

together and each layer is said to extract specific image features. The lower layers extract simple features like curves and edges while the deeper layers extract features that are more complex and individualistic to the specific image like facial features and expressions [14]. In recent years, deep convolutional networks have outperformed previous state-of-the-art techniques using ReLU activations and dropout regularization techniques when trained on the LSVRC-2010 ImageNet dataset making them the most popular architectures for vision related problems [15].

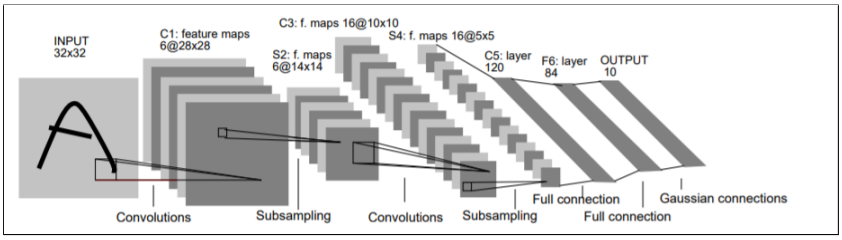


Figure 2.1.: Architecture of LeNet-5, a CNN used for digit classification [16]

- Recurrent Neural Network: Recurrent Neural Networks (RNNs), are neural networks that are well suited for temporal or sequential data like time-series or long textual data sequences. In a normal feed forward network, all inputs are considered to be independent of one another. However, for many language and time-series related tasks this idea would not work as the output is dependent on previous computations say for example predicting the next word in a sentence is dependent upon the previous words that came before it. This can also be expressed as saying that RNNs have *'memory'* as they store previous calculations that are useful for the future ones. However, in practice RNNs are limited to look back only till a limited number of steps. RNNs can be used to generate outputs after every time step or generate an output after reading a whole sequence making it suitable for different types of text classification, translation, captioning tasks. RNNs however can only go back upto a limited number of steps which is not useful in a lot of context dependent tasks, A special variant of RNNs called Long Short Term Memory (LSTM) have gained a lot of popularity in the recent years as they overcome this challenge of the standard RNN version. LSTMs are capable of learning long term dependencies by using a different structure consisting of four gates for different purposes.

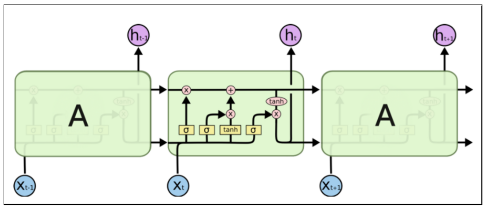


Figure 2.2.: Sequential LSTM blocks [17], [18]

2.1.2. Neural Networks: Methodologies

- One to One models: One to one neural networks are vanilla feed forward networks where there is a fixed sized input like an image or a vector which passes through some hidden layers and there is a single output which is based upon classification scores across certain labels or categories. Image classification tasks using deep CNNs are examples of such one to one models.
- One to Many models: One to many models on the other hand are neural networks that have a fixed size single input like an image and a varied size output like a text that serves as the caption for the image. Another example of such a network could be a single image classified into multiple labels or categories.
- Many to One models: Many to one models are neural networks that have a varied sized input like a vector representing a text and a single fixed sized output like the target label based upon computed classification scores. Sentiment analyses and text classification tasks are examples for such problems. An LSTM or a sequence model can be used for such problems with an output layer at the end to get the classifications.
- Many to Many sequence models: Sequence to sequence models are used in sequence based problems where the input and output sequence is not required to be of the same length. A sequence to sequence model consists of an encoder and a decoder with some intermediate units in between. This model tries to map an input sequence of fixed length to an output sequence of fixed length where both input and output are not necessarily of the same length. For text translation tasks, LSTMs would not work if the length of sequence in the source language is not equal to the length of the sequence in the target language. Sequence to sequence models address such problems.

2.2. Natural Language Processing

Natural Language is the form in which humans communicate with each other. Over the last few decades, many attempts have been made for computers to process natural language and understand it. NLP has evolved massively over the years, from taking several minutes to analyse words to the era of Google and its likes where millions of webpages can be analysed in a matter of seconds. It is a field of computer science which focuses on the interaction between human language and computers [19]. It usually involves large scale processing of natural language to obtain hidden patterns behind the textual information. This process can help in a lot of daily tasks like automation, text summarization, text categorization, translation, sentiment analysis and named entity recognition. A lot of enterprises today employ such techniques to automate and expedite the day to day tasks that get too monotonous for a human to solve. The NLP approach assumes natural language to be hierarchical in structure instead of mere symbols. Words result in phrases, phrases form sentences and a group of sentences have some semantic context. Thus, NLP enables extraction of meaning and

ideas from natural language which can be exploited and applied to numerous real world applications and problems that exist today.

NLP is not a straightforward technique to implement as language understanding is a very complex and convoluted task. It is difficult for humans themselves to understand ideas and meanings behind natural language. Natural Language is not precise or structured and hence difficult for computers to comprehend. There is what is called the ambiguity of language which makes NLP a difficult skill to master. Language ambiguity indicates that there is no precise or clear meaning to a sentence and it could possibly mean more than one thing. In speech and writing, there exists two types of ambiguity namely, lexical ambiguity and syntactical ambiguity. Lexical ambiguity indicates that there exists more than one meaning to a word whereas syntactical ambiguity means that a given sentence has two or more possible meanings. For a long time machine learning approach to solve NLP tasks have employed simple and shallow approaches that include high dimensional but sparse representations. In the last couple of years, deep learning has achieved much better results as compared to the traditional approaches. These Deep Learning approaches involve deep neural networks that take dense vector representations as input. There have been a lot of research to solve NLP tasks using deep learning. Many state of the art deep learning architectures have been built in the recent past to cope with NLP problems like entity recognition, text summarization, translation etc. Popular NLP problems are explained in the section below:

1. Part of Speech Tagging: Part of speech (POS) is a category of words that have similar grammatical properties. POS tagging is an approach of tagging each word in a given text input based on its lexical properties [20]. This problem is more difficult than it sounds as a lot of words can have different parts of speech when used differently in a sentence. The word *'out'* for example can have five POS tags depending upon how it is used in a sentence. This also varies from one language to another depending upon the ambiguity of a given language.

2. Dependency Parsing: This is a technique which involved grammatically analyzing a given sentence and building a parse tree. The parse tree is built depending upon relationships of words to one another such as defining the subject and predicate in a sentence. This is also not a straightforward problem given the ambiguity of natural language. Various combinations and solutions can be obtained for a given sentence. Stemming: Stemming is an approach which involves reducing a given word to its root word also known as the stem word. This reduces the size of text input massively where a lot of forms of a word are stemmed back to the root for example closing, closed, close, closer are all stemmed to the root word close. Stemming can help in a pre-processing text before applying further NLP solutions.

3. Named Entity Recognition : Named Entity Recognition (NER) is a technique that involves identifying entities such as person, organization, location etc. from a given text and labeling or mapping the entities to appropriate tags [20].

4. Text Summarization : Text Summarization involves summarizing a given paragraph of chunk of sentences. This is usually applied to sentences that belong to a single domain

or field only for example summarizing scientific journals on specific topics or generating headlines for newspaper articles or summarizing user reviews for a given product.

5. Machine Translation: Machine Translation involves translating a given text from one human language to another. This is one of the most demanded problems in today's world and is not trivial to solve as it requires syntactical, semantic and human like intelligence to understand and accurately translate text from one language to another.

6. Text Classification: Text Classification also popularly known as document classification is an approach which categorizes a given text into domains or categories based upon context and semantic meaning. The thesis work primarily aims at Categorizing the product information of a product listed on an e-commerce website into labels or classes like colour, brand, target gender and category of the product.

7. Sentiment Analysis: Sentiment Analysis involves extracting subjective properties of texts that classify them into generic and much wider classes like positive or negative labels for a given set of customer reviews. This approach determines the polarity of the text and is widely used in opinion mining for texts generated on social media, e-commerce and entertainment platforms.

8. Question Answering: This approach tackles the answering of a human generated question. This question could be straightforward or vague and the performance of the algorithm highly depends upon what kind of question has been asked. There have been algorithms developed in the recent past that can also provide with the accurate position of the answer in the given text at hand for the question that has been posed.

2.3. Computer Vision

Computer Vision (CV), is a field of Computer Science that seeks to develop techniques in order to enable computers to understand digital images like photographs and videos. The focus of CV is to enable computers to interpret something about the world by observing images. It could be considered as a sub-field of Artificial Intelligence that utilizes algorithms to perform vision related tasks. It is pretty similar to the human way of extracting information after observing images. This could be in the form of object detection, face recognition, image segmentation, motion tracking etc. The research in the field of CV started in the 1950s and in the last couple of years numerous advancements have been made in this field with convolutional neural networks (CNN) being one of the major ones. CNNs today can be used for complicated tasks like autonomous driving, object detection and facial recognition. Most of the vision related applications involve measurement and processing related tasks done by employing various methods. Some of the typical computer vision tasks that are applied to real world problems are mentioned below:

1. Image Recognition: Image Recognition is a technique in CV that tries to detect all the objects in a given image or video sequence. Image recognition can also be used as an

identification technique where individual characteristics of an image is detected and these characteristics are used to identify it in the future. This method can be used for facial recognition and fingerprint identification tasks. Another type of recognition task is detecting specific properties and scanning an image based upon a condition. This is particularly useful in detecting abnormal cells or tissues in medical imaging or vehicles on the road. Detection is a comparably faster technique as one focuses only on specific regions of the image to extract information. Currently, the best algorithms for such tasks are based on CNNs. Numerous state of the art architectures for image detection are trained on the ImageNet dataset which is a benchmark dataset in object classification and detection consisting of million of images belonging to thousands of classes. As easy as it sounds, algorithms still suffer in detecting and classifying objects due to distortions, background clutter and occlusion.

2. Image Classification: Image classification is a very popular problem in the field of Deep Learning. In this problem, a fixed size image is used as input and a fixed size output classifies the image by computing probability scores to determine the class of the image. The image is passed through a number of layers that could be fully connected, pooling, activation and convolutional layers. The ImageNet challenge is a popular image classification challenge that classifies images into 1000 categories and various models have been developed over the years to resolve this problem.

3. Motion Analysis: Motion Analysis involves studying methods and applications in which two or more consecutive images from a video sequence which is usually filmed using a high speed camera or a video recorder, are processed to extract information about the image. This could be based on three scenarios namely, the camera static and objects moving, the objects static and the camera moving and both the objects and the camera moving. One of the simplest ways of performing motion analysis is motion detection where points that are in motion are detected in an image with reference to previous frames in the sequence. More complex types of processing can be tracking a specific object in the image over time, to group points that belong to the same rigid object that is moving in the scene, or to determine the magnitude and direction of the motion of every point in the image. The extracted information tells something about specific objects in the image with reference to time.

4. Scene Reconstruction: Scene Reconstruction involves reconstructing a three dimensional version of one or more images or video sequences. The simplest way of doing it is by getting a set of 3 dimensional points using which the image can be reproduced or producing a complete three dimensional surface model.

5. Image Restoration: Image restoration involves removal of noise from a given image. Noise could be in the form of motion blurring or noise from sensors. The basic way of doing so is by using various types of low pass or median filters. More advanced techniques involve distinguishing the local structure of the image from noise. The local structures include curves, lines and edges. This could also be used in restoring damaged

images by reconstructing lost or deteriorated parts of an image or a video.

2.4. Neural Networks: Approaches

2.4.1. Single Task Learning

Single task learning (STL) is the most practiced approach in Machine learning where a single model is trained to solve a particular problem at hand. The desired metrics are optimized by fine tuning parameters until the performance of the model no longer increases. In such a scenario, the model is trained in isolation and the network works at approximating a function to get one output only. Single task learning is widely used all across. However, it has some drawbacks when it comes to generalization across multiple tasks. By focusing on one task only, often information that could be useful to the task gets ignored. This information usually comes from related tasks and can help in improving performance beyond what can be achieved from training the model in isolation.

2.4.2. Domain Adaptation

Domain Adaptation is a technique that allows knowledge from a source domain to be transferred to a related target domain. Recent works have also shown that domain adaptation techniques work well with unrelated source and target datasets. There are various ways of implementing domain adaptation techniques and we shall discuss a few of them in the sections below.

- Transfer Learning: Transfer learning (TL) is an approach which relaxes the assumption of the fact that the data in the training set and the test set need to be independent and identically distributed. This helps in solving one of the biggest problems in deep learning today, limitation of large training datasets. Transfer learning tries to transfer the knowledge from the source domain to the target domain which is the task in question. Transfer learning aims at improving performance of a task **T** belonging to a domain **Dt** by transferring latent knowledge from a task **S** belonging to a different domain **Ds** and usually trained on a much larger dataset[3]. The source domain task and the target domain task may or may not be same. However, the source domain and the target domain i.e. the feature space and the edge probability distribution are not same. Currently, transfer learning has become popular across NLP and computer vision related tasks where pre-trained models trained on large datasets like the Imagenet are frozen or fine-tuned to adapt to the target domain in order to improve performance. Transfer learning can be done in two major ways i.e. Fine-tuning the network or Freezing the network. The fastest and easiest way to do transfer learning is using a pre-trained model and freezing all its layers. The last layer which is where the classification happens is adapted according to the task in the target domain and then only the last layer of the model is trained using the new dataset which is usually much smaller. This involves no back propagation and as a result can be implemented very fast. Fine-tuning on

the other hand involves re-training of the parameters. However, the parameters are already initialized by the task in the source domain. A lot of the problems today are solved by a combination of freezing and fine-tuning of network layers [21]. The initial layers of a deep neural network, generally learn simple features that are universal across different domains. Therefore, these layers are frozen whereas the deeper layers which learn more complex features that are specific to only the task at hand are fine-tuned and re-trained. Which layers to freeze and which to train is a very important research topic today and is not a straightforward problem. A lot of trials and experiments are required to determine the layer from which the network should be fine-tuned. However, transfer learning is a favourite approach among a lot of researchers today because of its ability to overcome dataset shortage problem and achieve good performance. Transfer learning can be broadly categorized into four types. In this section, we shall briefly discuss each type.

1. Instance-based transfer learning: This is an approach where instances from the source domain are used as supplements with assigned weights in the training set of the target domain. This approach assumes that even though there are a lot of differences between the source and target domains, partial instances from the source domain can be used in the target domain using a weighting strategy that can supplement the training process in the target domain [3].
2. Mapping-based transfer learning: Mapping based transfer learning approach involves the mapping of instances from the source as well as the target domains into a new features space thus making the instances from the two domains more similar and suitable for a union neural network [3]. It is based on the assumption that though instances from both the domains are different, they can be similar in a new data space.
3. Network-based transfer learning: In this approach, partial network from the source domain is used and added to the network in the target domain. In this approach, the source network architecture and the connections are used as is in the target domain. This approach assumes that a deep neural network is somewhat similar to the extractive process of a human brain which tries to extract simple features in the first few layers and goes on to extract more complex features that helps the model to make the decision using the deeper layer [14]. Therefore, the partial network from the source domain can be used as a feature extractor and added in front of the target network. An example of such a scenario would be using the first few layers of a pre-trained network that has been trained to classify images belonging to Imagenet and transferring the trained layers as the feature extractors in any other visual recognition task like face detection, image segmentation etc [3].
4. Adversarial-based transfer learning: Adversarial based transfer learning approach involves in obtaining a representation that can be transferable from the source to the target domain and be of use to the latter. This is based on the concept of generative adversarial network where the model tries to learn the joint distribution of the data

in order to generate synthetic data which is very similar and almost indifferentiable from the source domain. For effective transfer to take place, good representation should be discriminative to the target task and indiscriminate between the source and the target domain.In this process the first few layers of the source network is used to extract features from the source dataset which is usually very large. The features from both source and target domains are sent to an adversarial layer which tries to see if the features are similar or far apart. If the adversarial layer is unable to differentiate between the two domains, then the transferability of the source domain is said to be high. During the training process, the adversarial layer forces the source network to discover more generic features that can be effectively transferred to the target domain [3].

Though transfer learning is a very popular choice when it comes to limited training datasets and getting good performance over a short period of time, it does not always work well and can even hurt performance rather than improving it. [3] Therefore, it is important to evaluate and be sure if at all transfer learning is a good option for a task at hand. In general, transfer learning is most likely useful in initializing the model better and more than often requires only refining the layers for the task at hand. The time taken for the model to converge is also much less as compared to when the model is trained from scratch. The model also converges much better as compared to models that are trained from scratch.

- Multi-task learning: Multi-task learning (MTL), is an approach that tries to overcome the limitations of training a model in isolation by promoting generalization across multiple tasks. This is done by sharing representations between related tasks. In contrast to single task learning, MTL involves optimizing more than one loss function and generalization across tasks by leveraging domain-specific properties from related tasks [22]. Even when the focus is on a single task, MTL using auxiliary tasks can help improve the performance of the main task. MTL is a form of inductive transfer where the related tasks provides an inductive bias to the main task to prefer certain hypotheses over others. The inductive bias coming from related tasks as a result make the main task choose hypotheses that generalize better across all the tasks rather than focus on the specifics of the main task only [23]. This method results in solutions that generalize better.

  MTL can be broadly classified into two types namely, hard parameter sharing and soft parameter sharing of the hidden layers [24]. Hard parameter sharing is one of the most popular approaches of MTL where the hidden layers are shared across all the tasks while each task has its own specific output layers. This approach primarily reduces overfitting as the model has to learn a representation that captures all the tasks instead of one specific task. This reduces the chance of overfitting on the main task. Baxter, 1997 showed that the chance of overfitting can be reduced by a factor of N if N tasks are trained in parallel using hard parameter sharing. On the other hand, soft parameter sharing is an approach where each task has its own set of parameters and each tasks

is trained separately using a model. The distance between the parameters across all the tasks is then regularized using L2 regularization methods or trace norm. [24] This ensures that the parameters are similar and promotes generalization. In the section below, we shall discuss what exactly makes such an approach work and how it can be used to overcome limitations such as dataset shortage and overfitting while training models in isolation.

1. Data Augmentation: MTL helps in implicit data augmentation because while training two tasks say A and B, one is able to average the noise patterns coming from both which helps in obtaining a better representation of the common hidden layers and results in generalization [24]. If task A was trained in isolation, the model would overfit to task A. However, MTL enables in ignoring the dependency on noise and can only learn useful patterns
2. Attention Focusing: MTL enables the task to focus on relevant features only while ignoring noise. Generally, the dataset at hand is complex and high dimensional. This results in tasks to learn irrelevant patterns that do not benefit the training process in anyway. By using MTL, the incorporation of other tasks helps in providing additional evidence for the relevance or irrelevance of features thus helping the model to focus its attention only on important patterns [24].
3. Eavesdropping: A lot of times, a task is unable to learn all features as it might interact with those features in a more complex way as compared to a different task. In such scenarios, MTL is beneficial as the task can depend on other tasks for learning features it itself find too complex to interact with. This method is called eavesdropping and can be done through the concept of hints [24] i.e. the model is trained to get the most important features.
4. Representation Bias: MTL provides an inductive bias forcing the model to prefer hypotheses that perform well for all the tasks in question rather than one single task only [24]. This way the model is able to generalize better as it is biased to learn representations that are preferred or that suit all the tasks. The model will be able to perform reasonable well for new tasks as well as long as they are from the same distribution.
5. Prevents Overfitting: MTL also acts as a regularizer as it prevents overfitting by providing an inductive bias [24]. Additionally, it also reduced the risk of the model fitting random noise and learning patterns that are not important.

In theory, MTL can be exploited to improve model performance because of the above stated reasons. However, the inductive bias can sometimes even hurt performance which is why MTL is not suited for all the applications in question. MTL could be a natural fit where we want to obtain multiple predictions for multiple tasks like in drug discovery where one wants to predict multiple symptoms of a disease. MTL could also be beneficial in scenarios where we are focused on a single task but can use auxiliary tasks to improve performance. This could be really useful in NLP applications where auxiliary tasks like tagging, chunking, dependency parsing etc. can improve

performance of language translation task. Therefore, it would be interesting to see if MTL can be useful for a given task and a specific dataset at hand or not especially in the e-commerce domain.

3. Related Work

This chapter covers the state-of-the-art techniques that are closely related to the involved topics of the thesis namely, transfer learning and multi-task learning. In the first section, we highlight recent works in transfer learning in various domains. The second section discusses application of multi-task learning across multiple domains and modalities. It describes MTL applications in e-commerce as well in the field of psychology. The related work in the transfer learning and multi-task learning section focuses on both NLP and CV applications.

3.0.1. Transfer Learning

Transfer learning is a very useful technique when a given dataset has insufficient samples for training. Transfer learning enables to transfer a model pre-trained on a source dataset to a target dataset. Though transfer learning can be useful in a lot of scenarios, there are cases when transferring knowledge could result in negative transfer and adversely affect the target training rather than improving it [25]. In the subsection below, we discuss scenarios where transfer learning has been applied.

- Transfer learning on Image based Datasets: Transfer learning on Image based Datasets: Amanda Ramcharan et al in their paper discuss the applicability of transfer learning in detecting Cassava disease detection using cassava disease images taken in the field of Tanzania [26]. The authors employ a state of the art deep CNN to identify three types of diseases and two types of pest damage. Cassava is a source of carbohydrate for humans especially in Africa. The outbreak of cassava virus has caused a threat to food security in regions of Africa since the 1990s. During the study, the authors prepared two sets of datasets, one called the original cassava dataset and second called leaflet cassava dataset which is manually cropped into individual leaflets [26]. The two datasets consisted of six manually annotated classes out of which three were diseases classes, two were mite damaged class and one healthy class. Each class consisted of not more than 500 images which is not enough to train a model from scratch. Therefore, the authors used the concept of transfer learning to leverage state of the art architectures and overcome the limited dataset problem. The authors used the state-of-the-art inception model called Inception V3 trained on the Imagenet dataset [27]. The model was tasked to classify images into a thousand categories. The last layer of the architecture is modified and retrained to classify the cassava dataset by exploiting the knowledge gained by training the model on the Imagenet dataset. The authors studied three approaches namely, using the original inception softmax layer, K Nearest Neighbours and support vector machines approach. The authors observed that employing transfer learning performed much better than randomly guessing despite varied backgrounds in the images like

sky, humans, feet, soil etc. It was observed that the accuracy ranged from 73 (for KNN) and 91 (for SVM) for the original dataset and 80 (KNN) and 93 (SVM) for the leaflet dataset [26]. The image classification task with TL from CNN Inception V3 is a powerful technique in this use case. It does not require training a model from scratch and saves a lot of time from feature extraction. Both SVM and the inception softmax layer perform pretty well as option for the last layer which is for the classification purpose. The results from the study show that a large dataset is not required when applying transfer learning on a target domain and that introduction of background clutter have little or no effect on the performance of the model. The authors aim to deploy such applications on phones to rapidly monitor such plant based diseases [26].

Another application of transfer learning in computer vision has been in the area of illustration classification. Garces and Lagunas in their paper, 'Transfer Learning for Illustration Classification' propose using models trained on natural images for the classification of illustrations and clip art data [28]. The authors initially create a baseline using VGG19 architecture which does not perform very well on the illustration dataset due to the differences in the imagenet and illustration dataset [29]. Next, they build two more models, one including VGG19 for extracting image representations and an SVM layer for making the classifications, second includes an optimized VGG19 along with an SVM. In the optimized approach, adaptive layer-based optimization of the network using the illustration dataset is done. The network is adaptively optimized layer by layer taking into account the differences of the target and source dataset.The optimized VGG19 produces in precision top-1 of 86.61% and precision top-5 of 97.21%. Improving the previous architecture by a 20% and 10% in precision top-1 and top-5 respectively [28].

Guo et al in their paper 'SpotTune: transfer learning through Adaptive Fine-tuning', propose an adaptive fine-tuning approach, called SpotTune, which finds the optimal fine-tuning strategy per instance for the target data [30]. In this approach, given an image from the target task, a policy network is used to decide whether to pass the image through the pre-trained layers or through the fine-tuned layers. The authors implement this approach on 14 standard image datasets and obtain improved performance on 12 of the 14 tasks. The authors discuss that though transfer learning is a very effective approach to obtain decent performance with limited data and in short amount of time, it is a matter of intense research to decide which layers of the pre-trained models to freeze and which ones to fine-tune especially when the model architecture consists of hundreds and thousands of layers. Some of the natural approaches include freezing the initial layers which learn simpler characteristics of an image like curves and edges and fine-tuning the deeper layers which learn specific features of an image like ears and tails or fine-tuning all the layers of the initialized network. These approaches do not seem to work when the target dataset size is extremely small as compared to the number of parameters to fine-tune and the model consists of thousands of layers. The authors discuss that a global fine-tuning strategy is not effective because there could be instances where a target task instance could me more similar to the source task dataset whereas

the others not as much and hence suggest to learn a decision policy for input-dependent fine-tuning. The policy is sampled from a discrete distribution parameterized by the output of a lightweight neural network, which decides which layers of a pre-trained model should be fine-tuned or have their parameters frozen, on a per instance basis [30]. The policy network is trained to output routing decisions (fine-tune or freeze parameters) for each block in a ResNet-50, pre-trained on the Imagenet dataset. During learning, the fine-tune vs. freeze decisions are generated based on a Gumbel Softmax distribution, which allows us to optimize the policy network using backpropagation. At test time, given an input image, the computation is routed so that either the fine-tuned path or the frozen path is activated for each residual block [30]. The authors compare the SpotTune method with other fine-tuning and regularization techniques using five datasets namely, CUBS, Stanford Cars, Flowers, WikiArt and Sketch, some more similar to the Imagenet than the other. The baselines include network with fine-tuning all layers, pre-trained network used as feature extractor, fine-tuning k layers, fine-tuning 50% of the layers at random, fine-tuning all the layers of Resnet-101 and using L2 based regularization technique for fine-tuning. The models are compared using evaluation accuracy. The results show that SpotTune performs better than all the approaches including Resnet 101 which has way more parameters to train achieving an accuracy of 92% on Stanford Cars dataset and 96.6 on Flowers dataset. The feature extractor technique performs the worst as it can reduce the number of parameters when applied to a new target dataset due to domain shift. L2 regulatization technique is second best to SpotTune. Manually deciding how many layers to finetune can impede performance. SpotTune differs from other fine-tuning techniques because it considers the similarity between source and target dataset for every instance of the target data and shares layers with the source task without parameter refinement which reduces overfitting and promotes better use of features extracted from the source task. The authors also used the SpotTune approach on the Visual Decathlon challenge and achieved new state of the art over ten datasets.

- Transfer learning on Text based Datasets: Tushar Semwal et al in their paper, 'A Practitioner's guide to transfer learning for text classification using Convolution Neural Networks', discuss the applicability of TL in NLP and explain how adjusting hyper-parameters and neural layers can enable positive transfer [31] .They compare results against state of the art approaches and provide best practices to achieve success in TL. The authors have experimented using five datasets namely, Amazon reviews, Yelp polarity, IMDb movie reviews, MovieReviews dataset and the Stanford Sentiment Treebank dataset, by extensively studying the ability of neural networks to transfer knowledge through empirical methods by segregating datasets into sources and targets based on compatibility. The bigger datasets are used as the source while the smaller ones as the target. The task at hand is that of text classification and the authors propose that one of the easiest way to do so is to initialize the target model with the pre-trained parameters trained on the source model and either freezing or fine tuning the layers in the target model. The authors adopted a variant of the CNN and initially trained it

on all datasets to prepare the baseline. The performance of the success of TL in NLP however is dependant on the semantic relatedness of the two dataset. A less similar dataset can produce better transfer result if the out of vocabulary words of the source dataset are considerably less and it has a good vocabulary size. The authors also show through experiments that transferring the embedding layer always results in positive transfer. Transferring the convolutional and fully connected layer might not always result in positive transfer and is highly dependant on the source dataset. The transfer of the output layer does not benefit the target dataset and more often than not impedes performance as it is specific to a particular task or a dataset. The authors compared the results to state of the art models like CNN-Kim, Dep-CNN, DSCNN-P and TE-LSTM and found that TL augmented CNN performs comparably as good with much less parameters to train.

Though convolutional networks work pretty well with images, performance is not as good when dealing with texts. Therefore different types of Recurrent Neural Networks like LSTMs are used when dealing with transfer learning for Text datasets. In the recent past, even advanced RNNs are not as good when compared with the Transformer which is a novel neural architecture that uses an attention mechanism which makes it much more suited for language understanding and modeling. In the paper, Attention is all you Need', Vaswani et al show that the Transformer outperforms both convolutional neural networks and recurrent neural networks in English to German and English to French translation tasks [32]. Besides better translation quality, the Transformer requires less computation power and is a much better fit for modern machine learning hardware, speeding up training by up to an order of magnitude [33]. Unlike RNNs that take numerous steps to make decisions that depend on words far away from each other, the Transformer applies a self-attention mechanism which directly models relationships between all words in a sentence irrespective of their positions. Apart from high accuracy and low computation requirements in comparison, the Transformer also visualizes how differently the network attends to different parts of a sentence while processing or translating a given word. This makes the Transformer well suited for Constituency Parsing tasks and the same network that is used for English to German Translation performs pretty well at this task as well.

Another massive achievement in the field of NLP in the last couple of years has been the introduction of ELMo, UlmFit and BERT. Conventional vector representations like Glove and Word2Vec do not take context into consideration while representing the word. ELMo looks at the entire sentence before assigning each word in it an embedding. It uses a bi-directional LSTM trained on a specific task to be able to create embeddings [34]. The bi-directional LSTM enables the model to get a sense of the next word as well as the previous word which is useful in getting the context of each word [35]. ELMo works very well because it has been trained to predict next word in a sequence of words, also known as Language Modeling. It therefore does not require labeled data as it can has large amounts of text data to perform next word prediction. ELMo can be used for transfer learning for text based datasets as a source network trained on a massive

dataset of the same language as the target dataset. The pre-trained model or the weights can be used as a component for other language related tasks or just as the embedding layer which takes context into consideration and hence performs much better. Howard and Ruder introduced 'Universal Language Model Fine-tuning' (ULMFiT), that can be used for transfer learning for any kind of NLP task [36]. ULMFiT is a language model trained on a massive corpus using AWD LSTM. It is analogous to the Imagenet dataset for transfer learning in CV where the ULMFiT model can be fine-tuned on a target NLP task in the same way. The model achieves great performance for target datasets that are as small of 100 samples per label for a binary classification task and the performance can be compared with a model trained from scratch on a 100 times larger dataset. ULMFiT outperforms the state-of-the-art solutions on six text classification tasks, reducing the error by 18-24 % on majority of the datasets [37].

Another milestone for TL in NLP was the introduction of BERT which is a model that outperforms state of the art solutions on a variety of language related tasks. The BERT model is a Transformer Encoder based language model that is conditioned on both left and right context [38]. Basically, BERT takes advantage of the ELMo and the Transformer models by using an Transformer model which is bi-directional. The BERT model has two versions namely BERT Base and BERT Large both consisting of a trained transformer encoder stack. The BERT Base is comparable to the Open AI Transformer [35] in size while the BERT Large is a huge model that achieves state of the art results. The model is trained for two tasks. The first one is a Masked Language Model which takes input in which 15% of the tokens are masked and the model tries to predict the masked tokens. The second task is to predict if the sentence is likely to follow another sentence. While the first task handles relationships among different words, the second task handles relationship between different sentences. The BERT model can either be used as a pre-trained model and fine-tuned to specific tasks like question answering, text classification, sentiment analysis and sentence tagging. Like ELMo, BERT could also just be used as the embedding layer and added to an existing model for language tasks such as a Named Entity Recognition (NER).

3.0.2. Multi-task learning

Multi-task learning is another very useful technique when one is dealing with dataset shortage problem. It can exploit task commonalities that leads to better generalization [24]. Multi-task learning enables a form of inductive transfer where tasks introduce an inductive bias that forces the model to choose more general hypotheses than others enabling the model to generalize better. Multi-task learning is an area of active research as it is not a very straightforward solution and requires a lot of study. Multi-task learning is being used in many domains as a solution to improve the performance of a particular model on a specific task by training it together with related tasks. In the subsection below, we discuss scenarios where multi-task learning has been applied and significant results have been achieved.

- Multi-task learning across multiple domain: In the paper, 'One Model To Learn Them

All', Kaiser et al propose a single model to perform tasks across multiple domains [39]. The model consists of neural network building blocks from multiple domains. These building blocks include convolutional layers, attention mechanisms and sparsely gated layers that form the MultiModel Architecture. The model is trained concurrently on 8 datasets namely, WSJ speech corpus [40], ImageNet dataset [2], COCO image captioning dataset [41], WSJ parsing dataset [42], WMT English-German translation corpus, the reverse of the above: German-English translation, WMT English-French translation corpus, the reverse of the above: German-French translation [39]. Through the experiments it was observed that the MultiModel performs similar for most of the tasks and even better for some tasks with sparse data in comparison with training each task independently: single task training. This indicates that tasks with small datasets like the parsing task benefit when trained jointly with large datasets such as the translation datasets. The parsing task has an accuracy of 97.1% when trained independently. The accuracy increases to 97.5% when trained together with the Imagenet task and to 97.9% when trained along with all the 7 tasks. However, the results achieved are not close to the state-of-the-art solutions for these tasks and the authors suggest that hyperparameter tuning could improve performance further. It was also observed that introducing unrelated computational blocks did not hurt performance of the unrelated task and sometimes even resulted in slight increase in performance. The Imagenet classification task has an accuracy of 67% with the attention and gated blocks. On removing the gated blocks, accuracy falls to 66% whereas on removing the attention blocks, accuracy remains at 67%. Therefore, suggesting that using blocks around multiple domains does not hurt performance for any task.

- Multi-task learning in the E-commerce domain: Wang et al in their paper, 'A Multi-task Learning Approach for Improving Product Title Compression with User Search Log Data', apply multi-task learning for improving the process of Product Title compression of products of an online e-commerce website using user search log data [43]. The authors train two models together. One being a sequence to sequence model with an attentive mechanism as an extractive method for product title compression and the other being an encoder-decoder approach to generate user search query given the original title of the product. The encoding parameters are shared among the two tasks and the attention distribution are optimized jointly. The authors focus on the task of online product title compression due to the massive growth in online users especially on mobile phones across numerous platforms including e-commerce websites. This trend has led to the need for improving user experience on mobile applications mostly because there is a massive difference between the screen size of a desktop computer and a mobile phone. The authors feel that there is a need for producing concise product titles which can fit and be readable on a small mobile phone screen. On various e-commerce platforms, millions of products are listed by external merchants that provide extensive product titles for search engine optimization purposes which results in bad customer experience. The authors use extractive summarization for title compression as compared to conventional text summarization techniques as merchants would less likely agree

to use words not existing in the original title especially if it reduces conversion rate [43]. Traditional methods of extractive summarization is time consuming and requires a large amount of data which is why the authors suggest using multi-task learning with user search log data. The multi-task learning framework consists of two attention based neural networks, one to model manually edited sort titles from original titles and the second one to model user search queries from original titles. Due to the absence of such a kind of benchmark dataset, the authors prepare the dataset from scratch from a Chinese e-commerce website crawling Womens' clothes category. The dataset consists of triplets namely, product title, manually compressed title and user search query with more than ten monthly transactions per month. The dataset consists 185k records with average length of product title being 25.1 characters, average length of manually edited titled being 7.5 characters and average length of search queries being 8.3 characters long. The authors use an encoder decoder network for both the tasks using attention mechanism. The encoder network consists of two 128 dimensional LSTMs and one 256 dimensional LSTM for the decoder. As the corpus is in Chinese language, character embeddings are used instead of words. The baseline models include a *Truncation based Method* which is a naive way of truncating the title after a threshold and an *ILP* based method which uses unsupervised learning by employing word segmentation, entity relationship and term weighting. The extractive summarization technique is achieved by using a pointer network to ensure that the words are not out of the vocabulary of the original title. For multi-task learning, two models are built. The first one where the final loss is the linear combination of the two sequence to sequence networks and the second one where attention distribution agreement based multi-task learning is used. Rouge score is used as the metric to compare the models. The authors also introduce a manual evaluation technique where 300 titles are sampled and three individuals are asked to evaluate the results based on 1) Core product recognition i.e. the product word exists in the title 2) Readability and 3) Informativeness. The Agreement based MTL performs the best on the basis of Rouge-L score as well as manual evaluation. The performance of ILP is far behind pointer based and MTL techniques in both the evaluation types. The authors even deployed the solution online and performed A/B testing on an e-commerce website of over 4 million users. They measured conversion rates and concluded that MTL improved conversion rates by a significant percentage as compared to the baseline hence concluding that MTL not only compresses titles for better user experience but also ensures improved conversion from click to a purchasing action [43].

- Multi-task learning for Mental Health in the field of Psychology: In the paper 'multi-task learning for Mental Health using Social Media Text', Benton et al model multiple conditions to make predictions about suicide risk and mental health using a deep learning framework [44]. As the experiments are in a medical setting, the authors ensure that the results maintain low false positive rate because predicting mental health risks for patients with no risks at all is very risky and is not practical. The authors use social media text produced by individuals suffering from mental health disorders

to predict individualistic attributes like age, gender, occupation, personality, mental health conditions and suicide risks. The authors feel that suicide risk is related to a lot of mental health disorders and therefore decide to use MTL approach to benefit from task commonalities and overcome limited training data problem. The authors propose that task selection is very important in order to attain good prediction performance and incorporation of auxiliary tasks like predicting gender also improves the model. The authors use twitter user dataset to get twitter posts of around 9000 users that are suffering from mental health conditions or have tried to commit suicides in the past. Each user as an average of 3500 twitter posts. The data is split into 1-5 grams at character level and fed to the model for ten text classification tasks including suicide, seven mental health disorders, neurotypicality and gender prediction. These tasks are trained separately using logistic regression and a feed forward neural network with two hidden layers. The same neural network with the same number of parameters is used for multi-task learning training where the first hidden layer is shared across all tasks and the last layer is specific to each task. The authors also try hyperparameter optimization techniques to obtain optimized weights and hidden layer size. The models are compared using ROC curve, AUC curve at TPR at FPR=0.1 to ensure that FPR is extremely low. The MTL model performs significantly better for all tasks as compared to STL and LR with TPR of 0.846 for neuroatypicality and 0.559 for suicide prediction. MTL also improved performance in predicting for tasks with relatively less data like for bipolar disorder and post trauma stress disorder. The authors also conclude that introducing an auxiliary task of predicting gender improves performance of other tasks though not necessarily improving the performance of the auxiliary task itself [44]. The authors also suggest using a perfect subset of the ten tasks to improve performance further rather than using all the ten tasks together. The results obtained show that MTL has a promising future moving forward. However, the current achieved accuracy is not good enough to apply the MTL models in a clinical setting.

4. Dataset

In the previous chapters we discussed the concepts on which the thesis is based upon and the applications of these concepts in varied settings. In this chapter, we shall outline the details about the datasets at hand and on which we shall experiment with the concepts of single task, transfer learning and multi-task learning. The chapter will then be followed by outlining the implementation of the experiments.

4.1. Benchmark Dataset

Through the thesis work, we try to solve the problem of missing product attributes of e-commerce products listed on a popular e-commerce website. Currently, there are no benchmark datasets publicly available that provide detailed attribute information of products currently listed on one of the e-commerce giants, Amazon. There however exists a public dataset in the e-commerce domain provided by Julian McAuley from University of San Diego, California [45] [46].

The dataset consists of approximately 142 million reviews by users on the Amazon.com website spanning from 1996 until 2014 [47]. This dataset is complemented with a product metadata dataset that consists product attribute information of 9 million products listed on Amazon.com. The categories include title of the product, category, price, brand, sales rank, image of the product etc [47]. Though the dataset seems very useful for the task at hand, it has some shortcomings. It does not contain information about currently listed products. The dataset also consists of products from the US website and therefore contains data only in the English Language. Moreover, the dataset does not contain detailed text attributes like product description, bullet points, product specifications etc. which will be very useful in performing NLP related experiments. Therefore, we have decided to create a similar dataset from scratch by scraping different regional websites in the European market by looking for products belonging to specific categories only. After fetching the list of products, we further scrape the detail page of each product to get all the required attributes of the product. The next section will provide further details of the scraping process which was the most fundamental stage in the dataset preparation phase.

4.2. Scraping the E-commerce platforms

The Scraping stage is one of the most fundamental steps of the thesis as the whole dataset is prepared by scraping the regional e-commerce websites. For the thesis, we have focused only on products that belong to the Fashion Category. We have scraped the UK, German,

French, Spanish and Italian Amazon regional websites for products that belong to the Fashion Category. The whole process of scraping the data was not an easy task. We took the help of a lot of literature available online that helped us through the process. One such article was by Hartley Body about important points to keep in mind while scraping huge quantities of data [48]. The set-up follows the important steps mentioned and provided in the repository [49]. Firstly, we retrieve navigation urls that belong to different sub-categories of the Fashion Department like 'Dress' , 'Men Shoes', 'Belts', 'Skirts' etc. From the sub-category urls, we retrieve attributes that include title of the product, detail page url, product image and price. Using this, we then use the detail url for every product to retrieve product level attribute information like product description, bullet points, product specification and download the image of the product. Though web scraping sounds straightforward as it involves making requests and extracting data from the response, it gets very complicated when one has to scrape for thousands of detail pages like in our case that too on a big website. Therefore in the section below we discuss the things that were kept in mind while developing the scraping process. First and foremost, when scraping a large and popular website, one needs to be aware that the website will be smart enough to detect bots due to vested interests in protecting information. In order to overcome this problem, we used Spoofing of headers and Rotating IP addresses using proxy servers. There are a lot of available services online that one could use to get proxy servers. In this case, we used Scraper API which is an online service that handles proxy management without the fear of getting blocked using a simple API call [50]. The services are provided at reasonable rates and the prices depend upon number of API requests and period for which service is required. Another important thing to note is the time spent on scraping thousands of web pages. The time spent on this could be really long which is why we need to do calculations to find out the time that we would require for 1 million requests. In order to boost performance and speed up things, multi-threading is key [48]. The crawler should be multi-threaded so that the CPU is busy working on one response or another, even when each request is taking several seconds to complete. According to the calculations, at least a couple of weeks to fetch all the required data was needed . Therefore, we used an EC2 instance from Amazon Web Services to run the crawler.

In this section, we shall talk about the library that was used to extract the HTML content from the web pages. Beautiful Soup is a python library that parses the content of a url and then traverses the retrieved parse tree to fetch anything that is required from it [51]. It also automatically converts incoming documents to Unicode and outgoing documents to UTF-8. Using this library we extracted required product attributes from the retrieved html content. The retrieved attributes included category, product description, bullet points, color, product specification, and image url of every product. We used a get method to fetch the image of the product from the image url and downloaded it. In this manner we were able to prepare three text based and three image based datasets in four weeks worth of time [52]. Please note that for text based datasets, we have data in German French and English language whereas for image based datasets, we have images from the UK, German and Italian websites.

Dataset	Input Sample	Label
UK Category	'Holstyle 0.6cm Heel Lift Half Insoles for Loafer....'	Insoles-Comfort
UK Color	'Mini dress with deep V-neck - Pink - 14-16....'	Pink
UK Brand	'Nike Slam Women's Dri-Fit Tennis Skirt - Black....'	Nike
UK Gender	'Cinda Baby Girls Christening Party Dress with Shoes....'	Baby girl
DE Category	'Rockabella Ivy Kleid schwarz/Weiss....'	Kleid
DE Color	'Damen Riemchen Abend Sandaletten High Heels....'	Schwarz
DE Brand	'Wrangler Herren Jeansjacke Auth Western:....'	Wrangler
DE Gender	'Sakkas Azalea Stein gewaschen gestickte Kunstseide Korsett....'	Damen
FR Category	'7 For All Mankind Bootcut, Jeans Femme....'	Jeans
FR Color	'Bloch Criss Cross, Chaussures de Danse....'	Noir
FR Brand	'Adidas FEF H JSY T-Shirt Homme Rouge....'	Adidas
FR Gender	'Chic Feet , Sandales pour femme 37.5....'	Femme

Table 4.1.: Table containing samples from the text datasets. Detailed samples can be found in A.1, A.3 and A.2

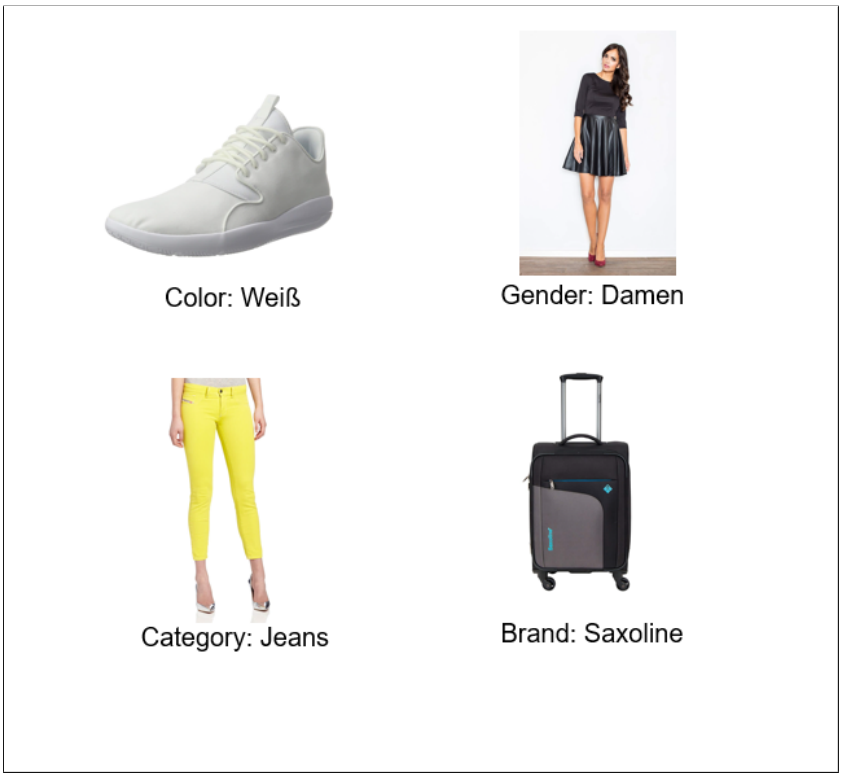


Figure 4.1.: Image samples for the 4 task types in the DE dataset.

4.3. Preprocessing

In this section we shall discuss the preprocessing steps that were involved in the preparation of text based and image based datasets for our experiments.

4.3.1. Text based Datasets

The various steps involved in the preprocessing of text based datasets are stated below:

1. Remove HTML Tags: As the data is HTML content, it contains a lot of HTML tags that is of no use to us. Therefore, we used the Regular Expression library to removed unwanted tags from the text [53].

2. Remove Special Characters: The data also contains some unusual characters like trademark signs, brand characters in languages other than the main language of the corpus. Therefore, we used the Regular Expression library to removed unwanted characters from the text [54].

3. Extract Gender from Category: The Category attribute scraped from the website contains a hierarchical tree structure like *'Women › Accessories › Belts'*. The structure contains the target gender the product belongs to. Therefore, we use some string manipulation techniques to check if the product is for the male, female, baby or the unisex gender. The dataset contains unisex items like suitcase, bags, keychains etc.

4. Extract the last element from the Category tree: The Category attribute is hierarchical in nature. Therefore, we extract the lowest element from the tree and use it as the category the product belongs to. For example, in case of *'Shoes Bags › Shoes › Women's Shoes › Court Shoes'*, we extract the last element which is *'Court Shoes'* and use it as the category label for this product.

5. Remove instances with empty label values: Not all the data scraped from the web pages consists of all the required information. Therefore for a given label, for example, brand, we remove all instances from the dataset that have empty brand value. As our task is a supervised machine learning problem, we will only deal with data that is labeled.

6. Remove labels with infrequent occurrences: In order to build a dataset that works well with machine learning algorithms, we remove labels that are a minority in the dataset. We remove all the labels and their instances for which there are less than 100 occurrences in the dataset.

7. Create the input feature for the model: The text input feature that will be fed into the model and classified into target labels is created by combing a number of text features that include product title, product description, bullet points and product specifications.

8. Truncate the input feature to a threshold length: The input feature is then truncated in order to ensure that all the instances are not more than the threshold value. This also

acts as a hyperparameter for the NLP model as the computation and time complexity will highly depend on this. Datasets with longer text sequences will take much longer to train [55].

9. Encode the text labels to integer values: The data is fed to a deep learning model and therefore the textual labels need to be encoded into integers for the model to understand it. Therefore, we use the Label Binarizer attribute from the Sklearn library in python to map string labels to fixed integer values [56].

4.3.2. Image based Datasets

The various steps involved in the preprocessing of image based datasets are stated below:

1. Prepare Folder Structure for each Label: The downloaded images are arranged in the respective label folders for each label type i.e. Brand, Category, Color and Gender. Each folder contains instances that belong to the target label and is named after the label instance.

2. Split the folder into training and validation folders: Each of the folders are then split into training and validation folders following a 95:5 proportion which means that the training set consists of 95% of the instances while the validation set consists of the rest.

3. Remove image instances that are corrupt or empty: During the scraping process it is very difficult to ensure that each image is in the usable format. Therefore, at this stage we try and get rid of all the corrupt and zero byte image instances that would be of no use in training the models with the dataset.

4. Remove labels with infrequent occurrences: In order to build a dataset that works well with machine learning algorithms, we remove labels that are a minority in the dataset. We remove all the labels and their image instances for which there are less than 100 occurrences in the dataset.

5. Resize the image instances: The images are then resized to be well suited for the deep learning architecture that we are going to use for the single task as well as transfer learning. In our case, we use the Inception-Resnet-v2 architecture and therefore resize the image to be 299 * 299 pixels.

6. Transform image instances: Using the transforms attribute from the torchvision library, the training images are flipped horizontally, and converted to tensors so that they are compatible with the model. The validation images on the other hand are center cropped as well [57].

4.4. Statistics: Text based Datasets

In this section we shall go through some detailed statistics of the datasets that we are going to conduct experiments on. The section is divided into two sub-sections, the first for text based

datasets and the second for image based datasets. In each of the sections, we shall depict the quantity of the dataset and the distribution of the class labels through tables and graphs.

4.4.1. Overview

Each of the dataset, has been prepared keeping in mind the target label for the task. For example, for the task of predicting category for UK, we used the input feature which was built using other text features like title, description, bullet points, specifications etc. and the category feature as input and target label respectively. We removed every record for which the target label was missing. Thus, each task has a unique dataset. The table below contains information for each of the text based dataset with the dataset size and the unique count of the target labels. As we can see the dataset consists a total of approximately 2.6 million records. The UK Gender is the dataset with the maximum number of records comprising of 14.8% of the total records followed by UK Category comprising of 14.5% of the total and UK Brand with 11.65% of the total. FR Gender comprises of about 11.3% followed by FR Category comprising of about 11% of the total, FR Brand with about 8% of the total and UK Color with 7.3% of the total. The DE dataset consists of the least number of records compared to UK and FR with DE Gender with about 5.7%, DE Category with 5.3%, DE Brand with 4.8% and DE Color with only 1.7% of the total dataset records. The proportion of each of the dataset is going to vital while conducting experiments for multi-task learning as a special parameter called threshold will be required to be set before executing the training for the tasks concurrently.

Dataset	Number of Records	Number of unique labels
UK Category	377,121	174
UK Color	190,538	154
UK Brand	303,967	599
UK Gender	387,594	8
DE Category	138,696	103
DE Color	43,343	64
DE Brand	125,362	269
DE Gender	148,090	8
FR Category	284,048	174
FR Color	111,681	120
FR Brand	211,101	498
FR Gender	293,659	6

Table 4.2.: Table showing the statistics of the text based datasets.

4.4.2. Input Length

The length of the input feature is a significant factor on which the performance of any NLP model depends. Therefore, picking appropriate length of the input feature in terms of number

of words is very crucial. The figures below visualize the distribution of the length of the input feature for each of the 12 text based datasets across UK, DE and FR. As can bee seen from the figures, majority of the occurrences have around 200-300 words as the input size. Therefore, for all the text based models, we shall use the maximum sequence length to be 300 and truncate every input feature to have 300 words only.

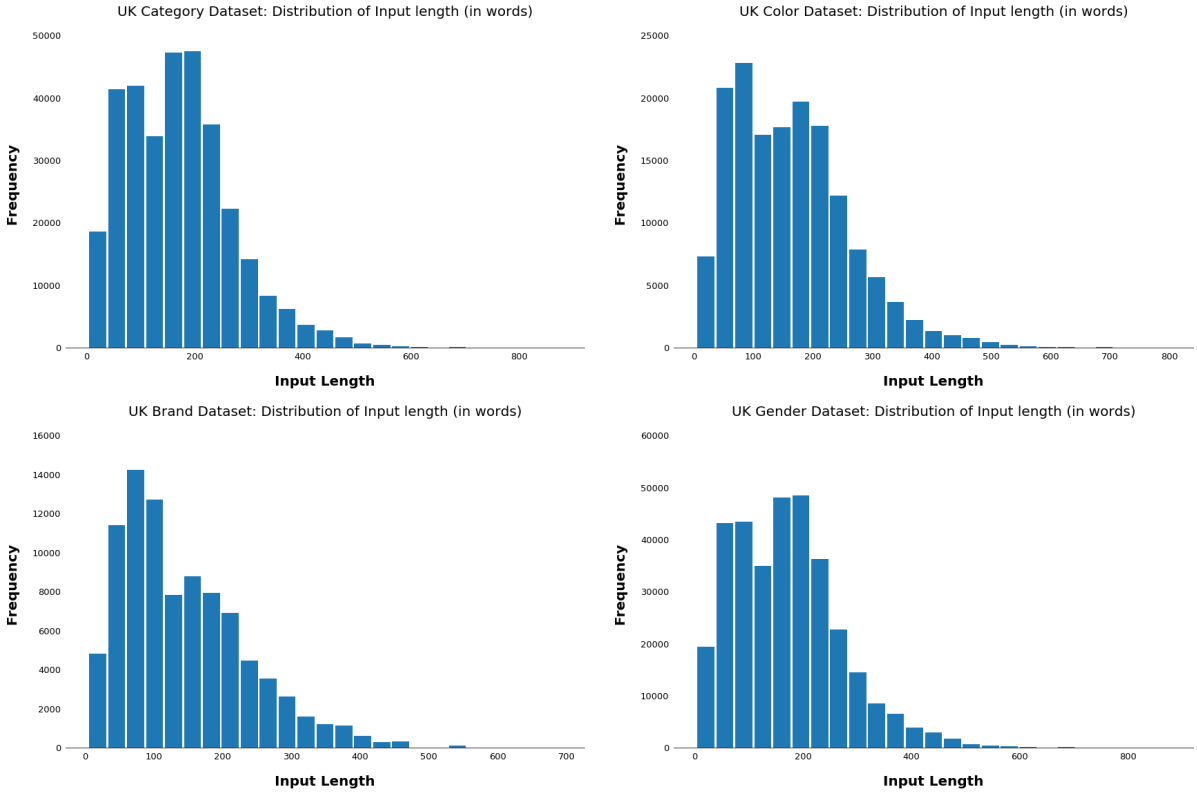


Figure 4.2.: Distribution of the Input length size for the 4 task types in the UK text datasets.

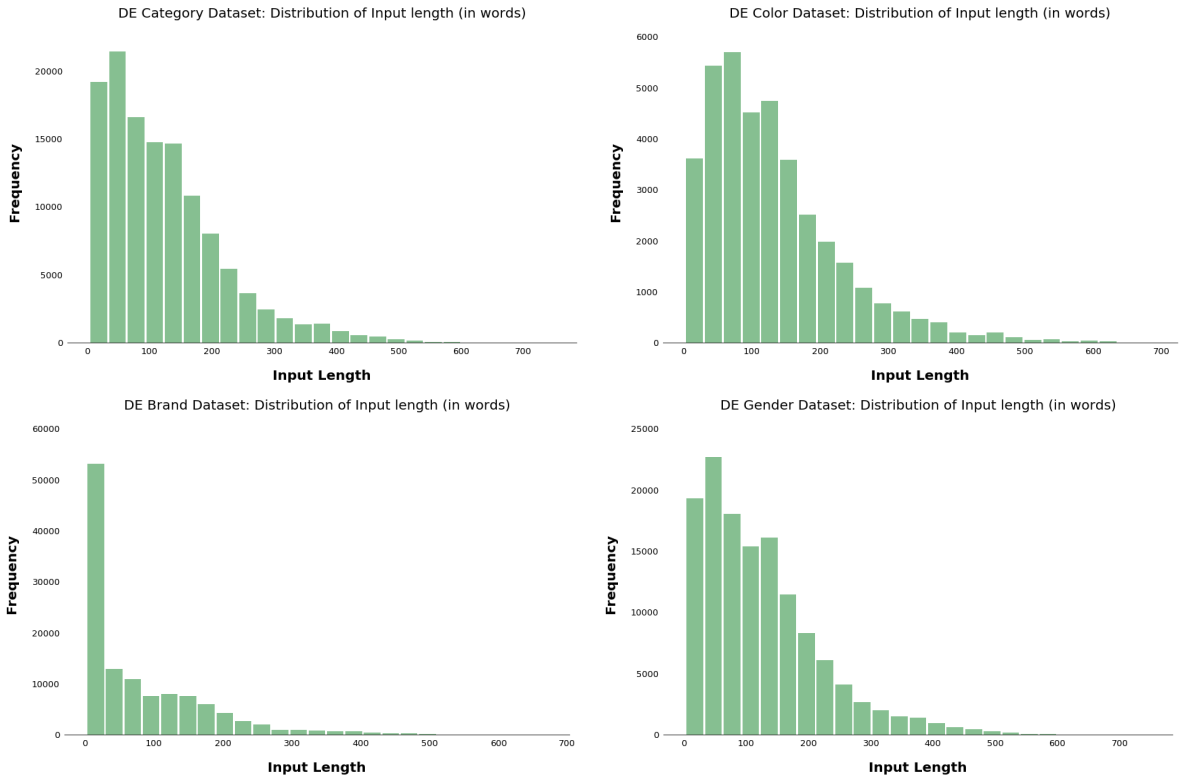


Figure 4.3.: Distribution of the Input length size for the 4 task types in the DE text datasets.

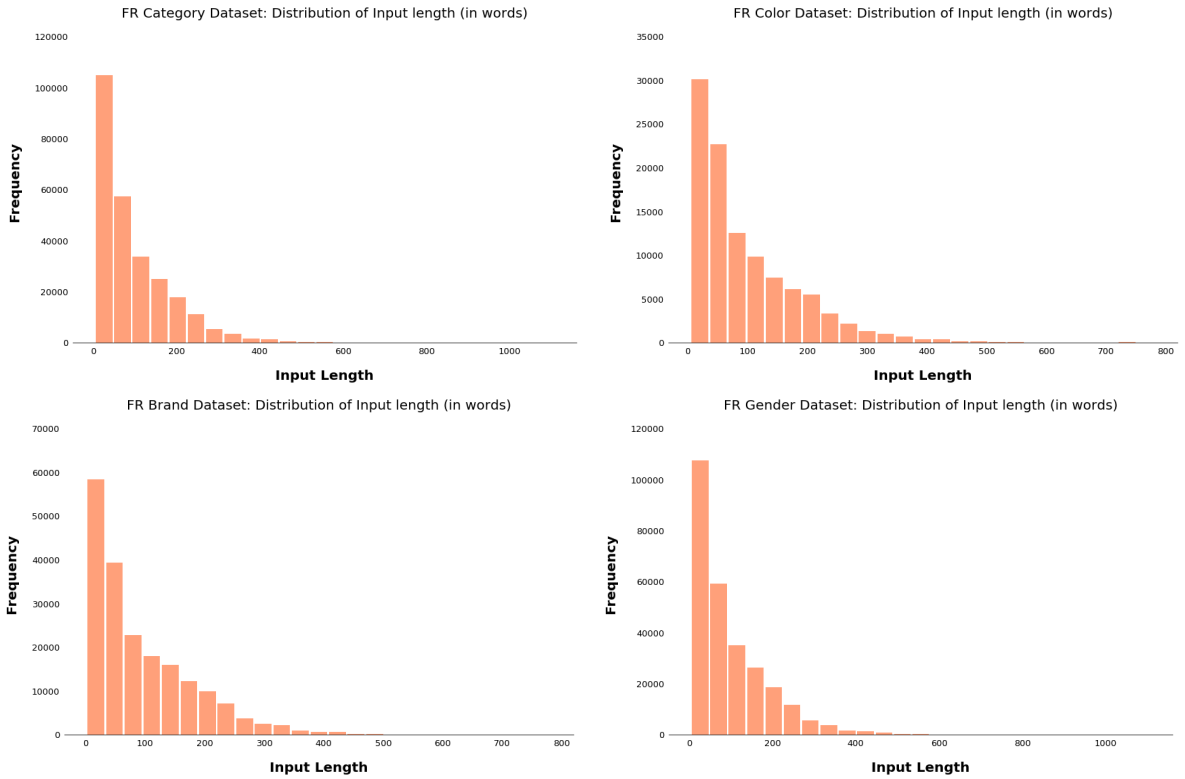


Figure 4.4.: Distribution of the Input length size for the 4 task types in the FR text datasets.

4.4.3. Target Labels

The figures below depict the distribution of the top 50 class labels for all the text based datasets. As can be seen from the figures, the dataset is imbalanced. This means that each of the class label does not have equal number of occurrences. Moreover, there are numerous labels in the range of a few hundreds for most of the task types i.e. color, category and brand. Therefore, we shall implement models that can tackle the unbalanced class problem and also deal with dataset shortage. This will be a major focus in our thesis where we shall employ domain adaptation techniques like transfer learning multi-task learning and compare the results with models trained independently without any knowledge transfer or sharing. Moreover, we would want to ensure that the model not only makes correct predictions for the majority classes but also for the minority classes. Therefore, we shall train the model using f1-score as the evaluation metric along with accuracy.

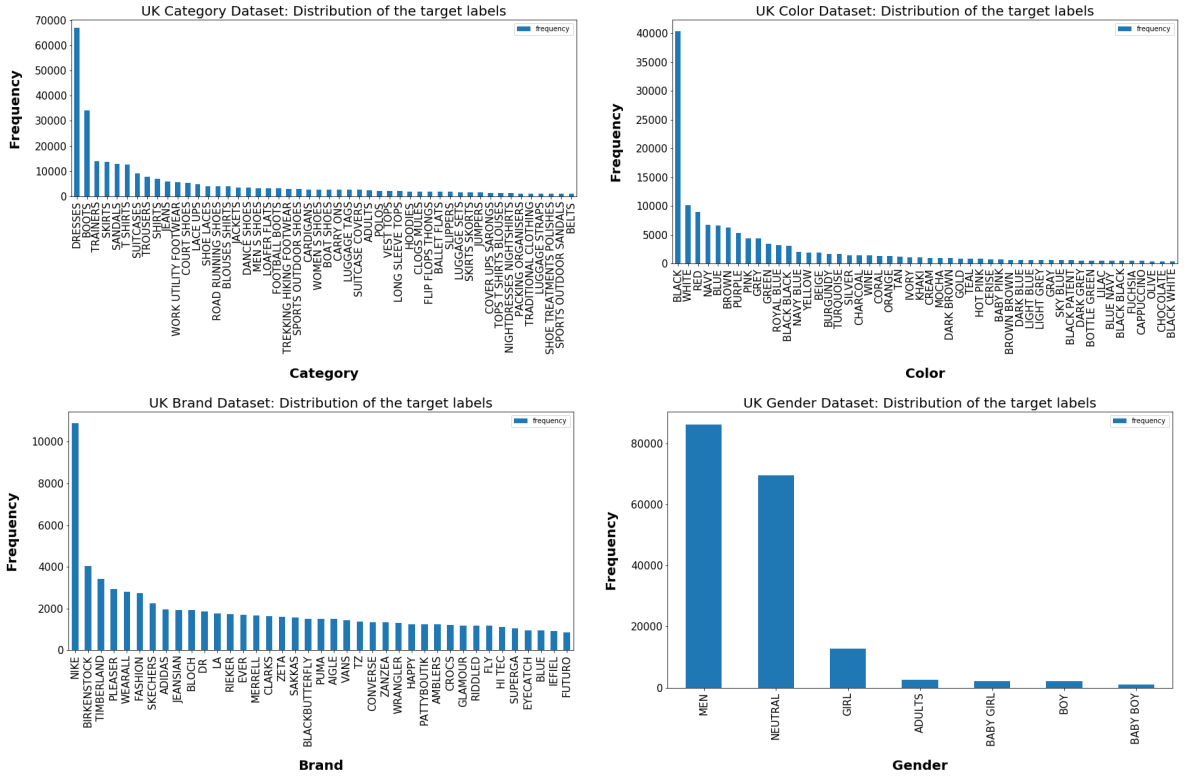


Figure 4.5.: Distribution of the target labels for the 4 task types i.e. Category, Color, Brand, Gender, in the UK Dataset.

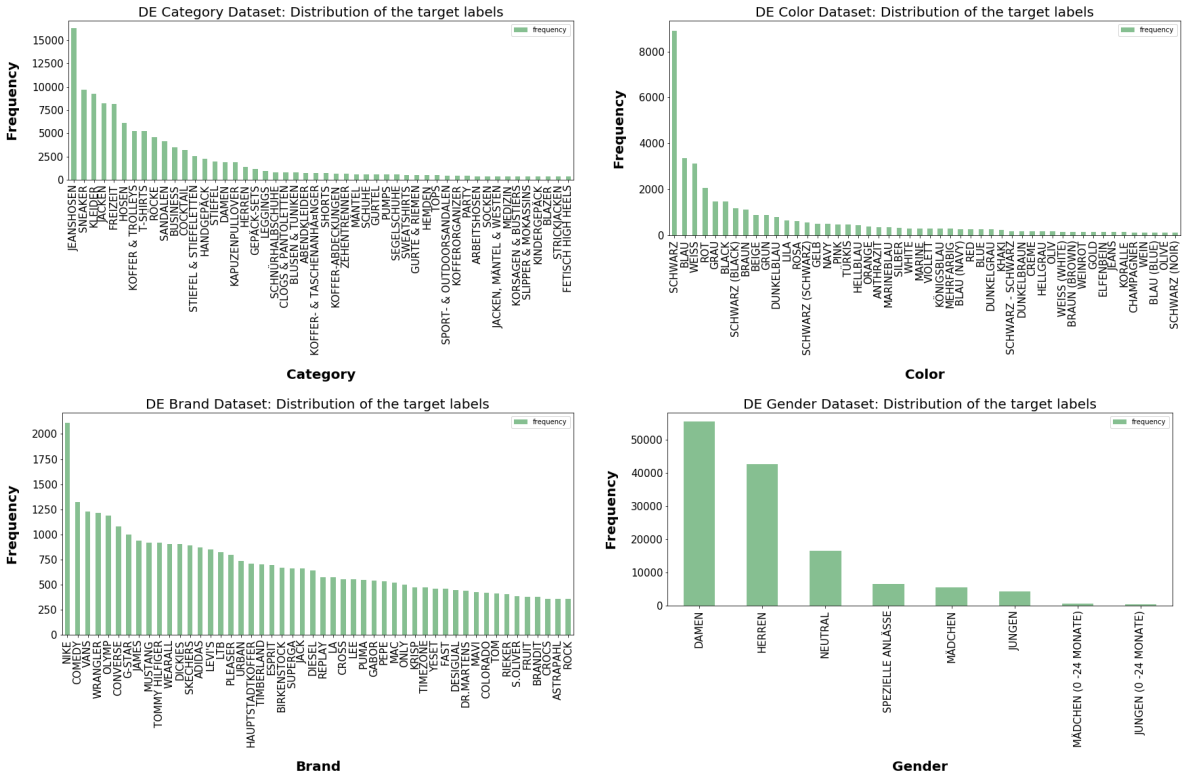


Figure 4.6.: Distribution of the target labels for the 4 task types i.e. Category, Color, Brand, Gender, in the DE Dataset.

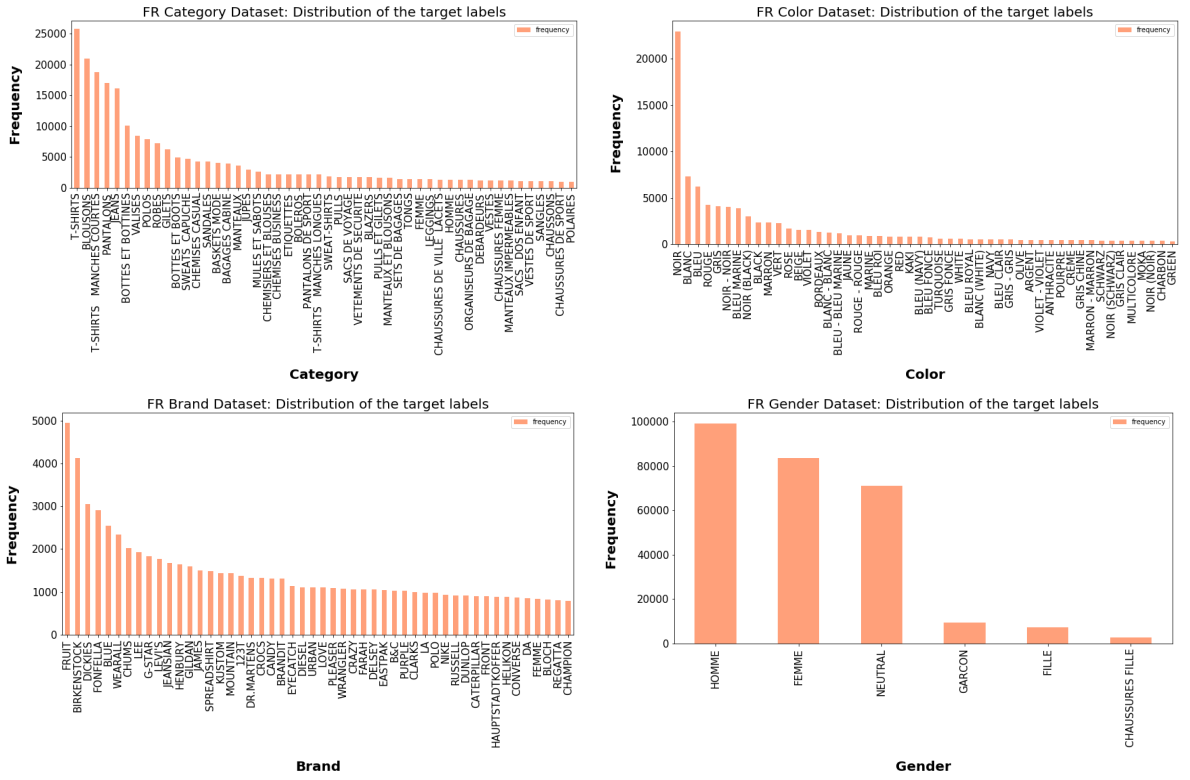


Figure 4.7.: Distribution of the target labels for the 4 task types i.e. Category, Color, Brand, Gender, in the FR Dataset.

4.5. Statistics: Image based Datasets

- Overview: The image based dataset consists of approximately 1.1 million images from the UK, DE and IT websites. The dataset with the most number of occurrences is the UK Gender dataset comprising of 10.2% of the total dataset followed by DE Gender and IT Color with 9.3% and 9.2% of the total dataset. This is followed by the UK Brand, DE Category and the DE Brand datasets with approximately 9% of the total. The rest of the datasets comprise of approximately 7-8% of the entire image dataset. The Color dataset for DE contains the least amount of data comprising of only 6.5% of the dataset. It will be interesting to see if it performs well when we conduct experiments using image based transfer learning in this case. Overall, the dataset is imbalanced with a number of majority and minority classes making the scenario ideal for domain adaptation. The table below contains information for each of the image based dataset with the dataset size and the unique count of the target labels.

Dataset	Number of Records	Number of unique labels
UK Category	82381	223
UK Color	96203	627
UK Brand	99728	172
UK Gender	113463	8
DE Category	98320	550
DE Color	74846	121
DE Brand	99564	197
DE Gender	102743	7
IT Category	89286	107
IT Color	101218	458
IT Brand	80458	145
IT Gender	87637	5

Table 4.3.: Table showing the statistics of the image based datasets.

- Target Labels: The image based datasets are very similar to the text based ones. The classes are imbalanced with a high number of unique classes for color, brand and category. Gender however has fewer number of unique classes for both text and image based datasets compared to the others. Due to the imbalance classes and limited amount of data available for each of the task, we shall use f1-score for the convergence of the models rather than accuracy. This will take care of the class imbalance problem. The images below depict the distribution of the various class labels for the three regional image based datasets.

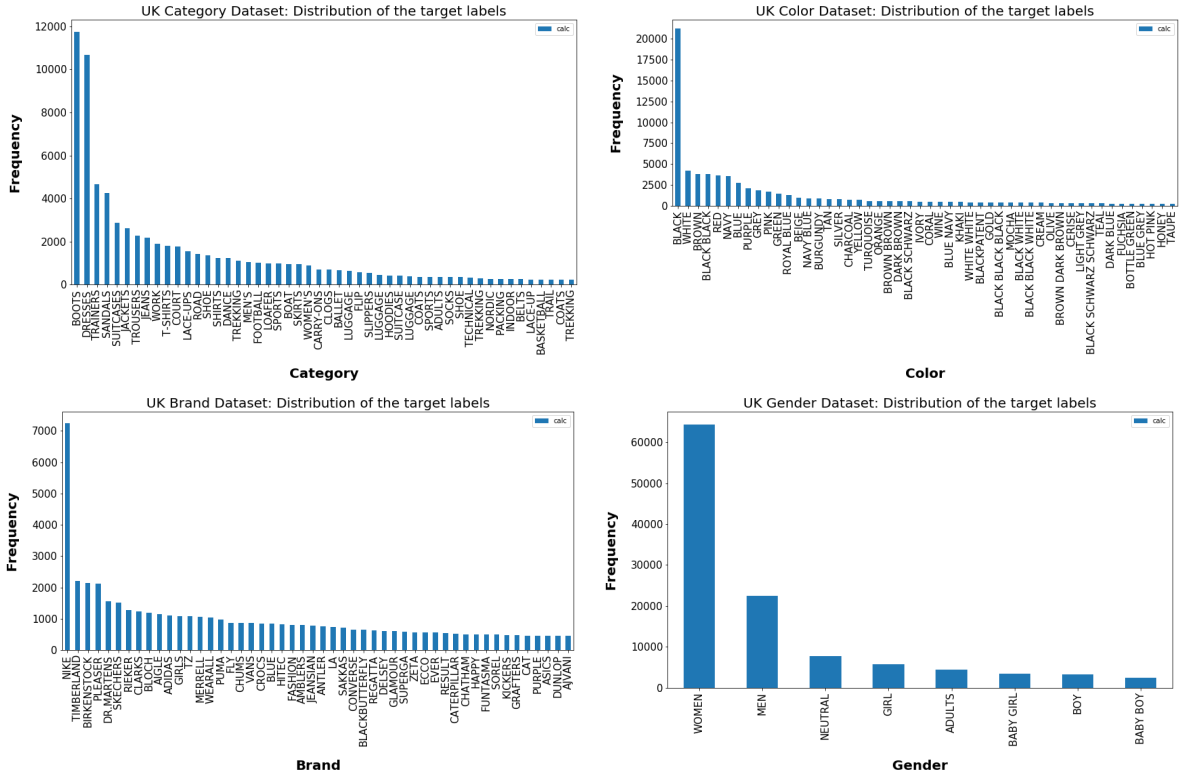


Figure 4.8.: Distribution of the target labels for the 4 task types i.e. Category, Color, Brand, Gender, in the UK Image Dataset.

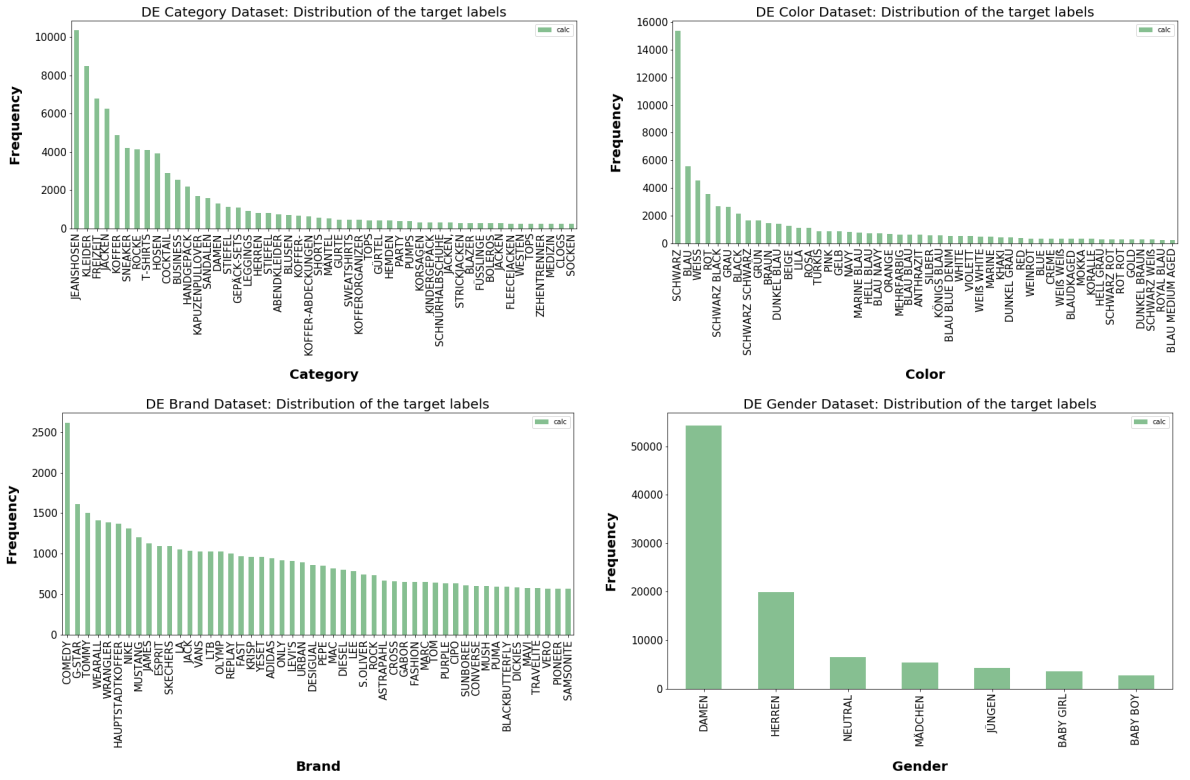


Figure 4.9.: Distribution of the target labels for the 4 task types i.e. Category, Color, Brand, Gender, in the DE Image Dataset.

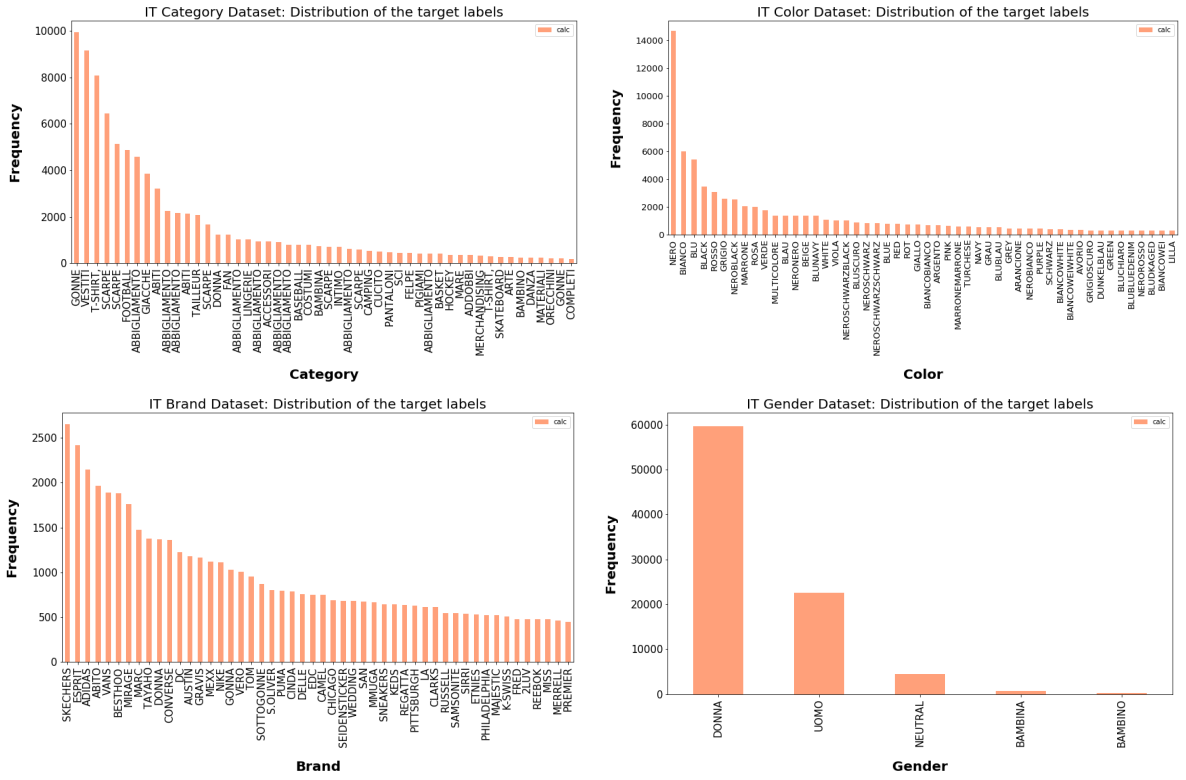


Figure 4.10.: Distribution of the target labels for the 4 task types i.e. Category, Color, Brand, Gender, in the IT Image Dataset.

5. Approaches

In this chapter, we shall discuss the methodology that was used in conducting experiments for each of the approach namely, single task, transfer learning and multi-task learning. Each of the section will explain the process of integrating the dataset with the existing framework, the algorithm and architecture used and outline the process of implementing each kind of approach for both text and image based datasets.

5.1. Single Task Learning

In order to compare the results achieved from transfer learning and multi-task learning, we have created a baseline using single task learning. In this way, we can conclude accurately if at all domain adaptation approaches are comparable or even better than the baseline model. We have outlined the process of implementing single task learning for both text and image based datasets in the sub-sections below.

5.1.1. Text based datasets

In order to train the datasets on classification tasks in three different languages, a model that can perform particularly well on text based datasets needs to be applied. Therefore, we use the state -of-the-art architecture for single task learning that uses an attention mechanism and works well with text related tasks [32]. This model is called the Transformer. In the paper, 'Attention is All You Need', Vaswani et al propose a simple network architecture based solely upon attention mechanisms. In the section below, we outline the architecture of the Transformer model and describe how training using this model can be easily implemented using the Tensor2Tensor (T2T) framework [58].

- Data Generation: The data generation phase is one of the initial phases of the single task training process. During this phase, the raw data is converted into TFRecord file format so that it can easily be interpreted by the model. During this phase, the vocabulary of the dataset is also created at subword or token level. The data generator in our case inherits from the Text2Text Problem class in which the vocabulary, evaluation metrics and the input and label pairs are generated. The dataset is also split into training and validation sets based on a defined proportion. The data generator scripts reads a text file into a pandas dataframe and generates the input and target label pairs. The labels are encoded into integer values so that they can be fed to the deep learning model. For each of the tasks and each of the languages, a unique problem is registered. Each of the

problem is then trained using the generated data, the Transformer model and a set of hyperparameters.

- Architecture: Architectures like the ByteNet [59] and the ConvS2S [60] use convolutional networks as the basic building block in the architecture because of which the number of operations required to relate signals from two arbitrary input or output positions grows in the distance between positions, linearly for ConvS2S and logarithmically for ByteNet [32]. The number of operations is reduced in the Transformer as it uses an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence. This is also known as self attention [32]. The Transformer consists of an encoder-decoder structure where the encoder maps an input sequence of symbols to a sequence of continuous representations and the decoder decodes this continuous sequence to an output sequence of symbol representations, one element at a time. Both the encoders and decoders consist of stacked attention and fully connected layers. The architecture of the network comprises of the following units:
  1. Encoder: The Encoder consists of six identical layers stacked together. Each layer consists of two sub-layers, one of which is a multi-head self attention layer while the other is a simple point-wise feed-forward layer. There is a residual connection around the sub-layers followed by a layer of normalization. The outputs of all the sub-layers have a dimension of 512.
  2. Decoder: The Decoder is identical to the Encoder apart from the fact that it consists of an additional sub-layer which performs multi-head attention [61] over the output of the encoder stack. The Decoder uses a masking mechanism and an offset by one position in the output embeddings to ensure that the output at position i only depends upon known outputs at positions before i itself.
  3. Attention:An attention function can be described as mapping a query and a set of key-value pairs to an output,where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key [32]. The Transformer uses a multi-head attention [61] mechanism which allows the model to jointly attend to information from different representation sub-spaces at different positions. The multi-head attention mechanism is used in three different ways. Firstly, it is used in the encoder-decoder layer to attend all positions of the sequence. Secondly, as self attention layers in the encoder to attend to all positions in the previous layer of the encoder. Lastly, it is used in the decoder as self attention layers to attend to the output of the previous decoder layer. To ensure auto-regressive property, all values in the input of the softmax which correspond to illegal connections are masked.
  4. Feed-Forward: Each of the layer in the Encoder and Decoder consists of a feed-forward sub-layer which is independently applied to each position of the sequence. This layer consists of two linear transformations and a ReLU activation.

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

5. Softmax and Embedding: The Transformer uses learned embeddings to convert input and output tokens to vectors of dimension 512. The decoder output is converted into predicted next-token probabilities using learned linear transformation and a softmax function. The weight matrix is shared between the learned embedding and the softmax layer.

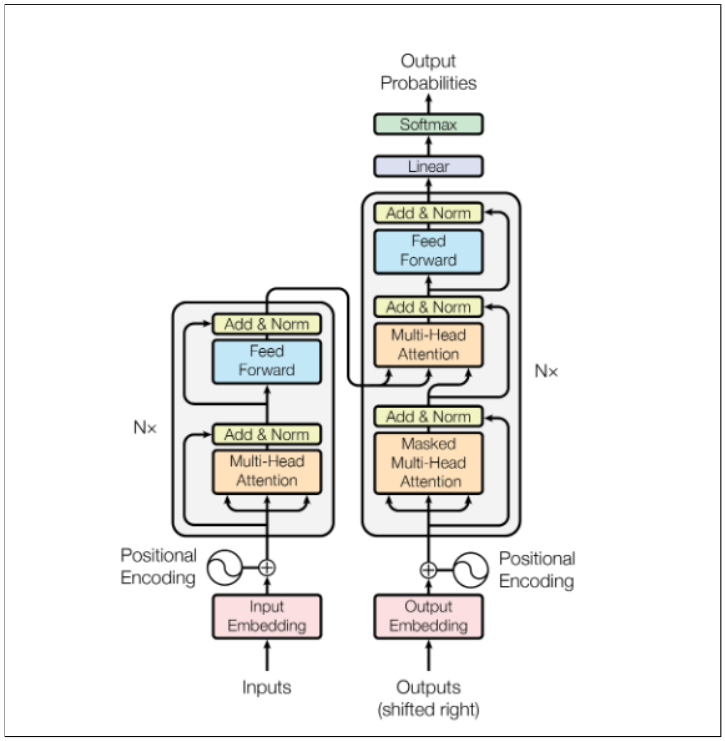


Figure 5.1.: Model Architecture of the Transformer. [32]

- Training: After the data generation phase, the TFRecord file is fed to the Transformer model in batches. Checkpoints of the model are saved after a defined number of steps which contains the current updated parameter values of the model. The model is periodically evaluated using these checkpoints on the test data. The evaluation metrics are monitored and the training process is stopped once the selected metric i.e. f1-score converges indicating that the performance of the model cannot be improved further. The metrics are monitored easily using tensorboard.
- Inference: After the model has been trained, the final checkpoints of the model can be used to classify unseen text samples into the various target labels. The unseen samples can be fed to the model in a file format. Using the Decoding step using T2T, the model provides with the outputs for each of the instance in the file.

5.1.2. Image based datasets

As discussed in the previous chapter, deep convolutional neural networks perform well in image recognition tasks. In order to train the image datasets on classification tasks, we use state-of-the-art Convolutional Neural Network that performs exceptionally well on the ILSVRC image classification benchmark [2]. The model is called the Inception-Resnet-V2. It is a variation of the InceptionV3 and borrows architectural ideas from the ResNet model using residual connections [62]. Szegedy et al in their paper propose a model which is a combination of the Inception architecture and residual connections [27]. The model is more accurate than the previous state of the art models improving top-1 and top-5 accuracy by almost 2% on the ILVRC benchmark dataset. In the section below, we outline the architecture of the Inception-Resnet-V2 model and describe how training using this model can be implemented using Pytorch deep learning library [63].

- Data Integration: The downloaded images are arranged in a definitive folder structure where each image is placed in the folder having the target label as its folder name as explained in the Datasets chapter. Each of the image is then cropped and resized in order to make it compatible with the model using the torchvision transforms module. The images are then converted from numpy image arrays to torch tensors. The torch dataloader module is then used to load and shuffle the data in batches. It also enables data loading in parallel by using multiprocessing workers. Every time the data loader is called, it iteratively loads images in the form of tensors based on the arguments defined.

- Architecture: The Inception-Resnet-V2 is a hybrid version in which residual connections are introduced that add the output of the convolution operation to the input. In order to develop residual versions of the inception module, 1X1 convolutions are used after the original convolutions. This ensures that the dimensions of the input and output after the convolutions remain the same. This is required to make up for the dimensionality reduction stage introduced in the Inception block. The computational cost of this architecture matches approximately with that of the Inception V4 network. The network, unlike its non-residual version, uses batch normalization only on top of the traditional layers and not on the summations. This enabled the possibility of increasing the number of inception blocks and training the model on a single GPU. The residual activations were scaled down by a factor ranging between 0.1 and 0.3 in order to prevent the network from dying when using deeper networks with number of filters exceeding 1000 [27].

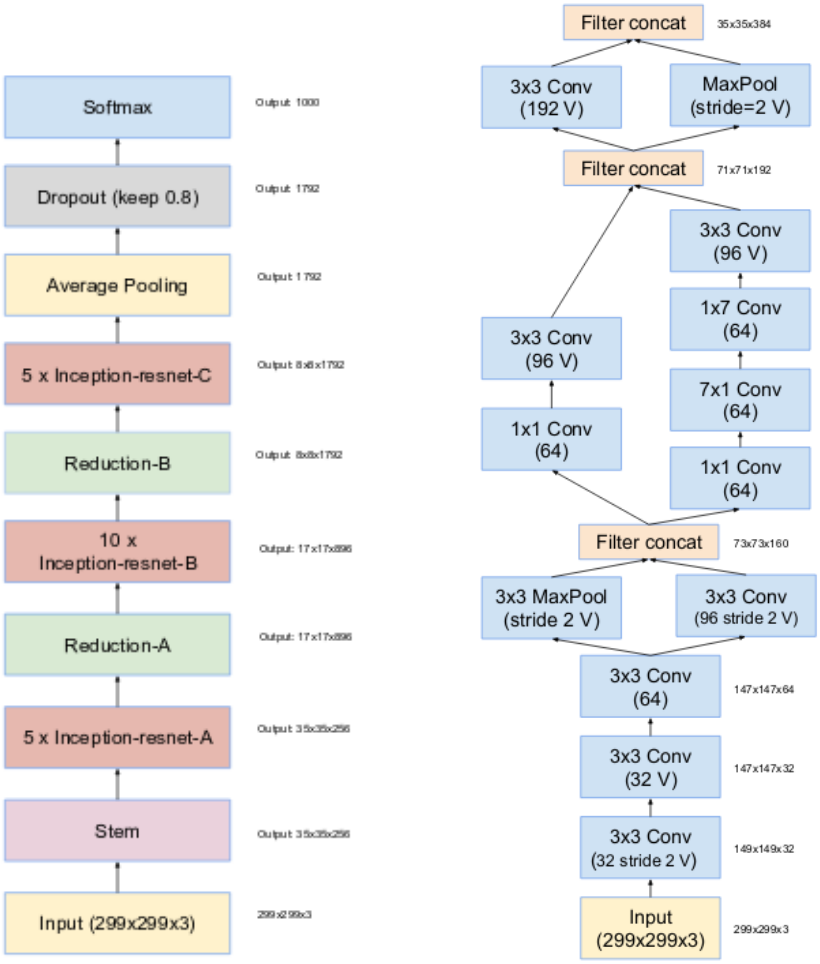


Figure 5.2.: Left: Inception-ResNet-v2 network architecture. Right: Schema of the Stem block in Inception-ResNet-v2.

- Training: After the data generation phase, the data loader is used to load training and validation image tensors in batches. Cross Entropy loss is used as the loss function and Stochastic Gradient Descent (SGD) with momentum is used as the optimizer during training. Checkpoints of the model are saved after a defined number of steps which contains the current updated parameter values of the model. The model is periodically evaluated using these checkpoints on the test data. The evaluation metrics are monitored and the training process is stopped once the selected metrics converge indicating that the performance of the model cannot be improved further.
- Inference: After the model has been trained, the saved checkpoints of the best model can be re-loaded to classify unseen image samples into the various target labels. The unseen samples can be fed to the model in the same way as was doing with the training and validation samples using transforms and data loader.

5.2. Transfer Learning

In the section below, we shall discuss the methodology employed to implement transfer learning on both text and image based datasets. Transfer learning can be easily implemented on images by loading state-of-the-art pre-trained convolutional networks. Previously, experiments had shown that transfer learning does not work as well on text based datasets as compared to images [64]. With the development of context aware networks like ELMo and BERT, transfer learning in NLP has gained massive popularity. A recent trend in transfer learning from language models (LMs) is to pre-train some model architecture on a Language Model objective before fine-tuning the same model for a supervised downstream task [65]. We shall implement pre-trained BERT base model to re-train it on text datasets. Similarly, we shall use pre-trained Inception-Resnet-v2 model and re-train it on the image datasets.

5.2.1. Text based datasets

- Data integration: At this stage, the raw text data along with the target labels are converted into a format that is interpreted by the BERT model. The BERT model is available as a loadable module through Tensor Hub and therefore we shall customize the existing text pipelines to integrate our datasets. Firstly, the InputExample class from BERT's run classifier code is used to create examples from the text data. Next, the data is preprocessed to match the data BERT was originally trained upon. The data is converted into lower case followed by converting the sequence into tokens (i.e. "sally says hi" becomes ["sally", "says", "hi"]). The words are then broken into WordPieces (i.e. "calling" -> ["call", "ing"]). The words are then mapped to indexes using the vocabulary file provided by BERT. Next, we add special "CLS" and "SEP" tokens to indicate the beginning or a termination of a phrase or sentence. The sequence is then truncated to the maximum sequence length depending upon the speed and memory capabilities available, 300 in our case. The run classifier_convert examples to features is executed on the InputExamples to convert them into features BERT understands [52].

- Architecture: Devlin et al in the paper, 'BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding', introduce a fine-tuning based approach by proposing BERT: Bidirectional Encoder Representations from Transformers. BERT applies bidirectional training of the Transformer for language modeling. It is a multi-layer bidirectional Transformer encoder based on the implementation described in Vaswani et al. [32] In our experiments, we shall use the base version of BERT which consists of 12 layers, 12 self attention heads and a hidden size of 768. The bidirectional nature enables the model to read entire sequence of words at once. A sequence of tokens are fed to the model which are embedded into vectors and then processed by the model. BERT is trained using two strategies. The first one involves prediction of words in the sequence that had been masked when fed as input and the second one being the prediction if the second sentence in the pair is the subsequent sentence in the original document. After training, the model can be fine-tuned and used in a variety of

NLP tasks like classification, question and answering and named entity recognition. In our case, we shall use the base version of the BERT model for classification. Most of the hyper-parameters stay the same as in the original BERT training, and only a few specific ones require tuning. [65]

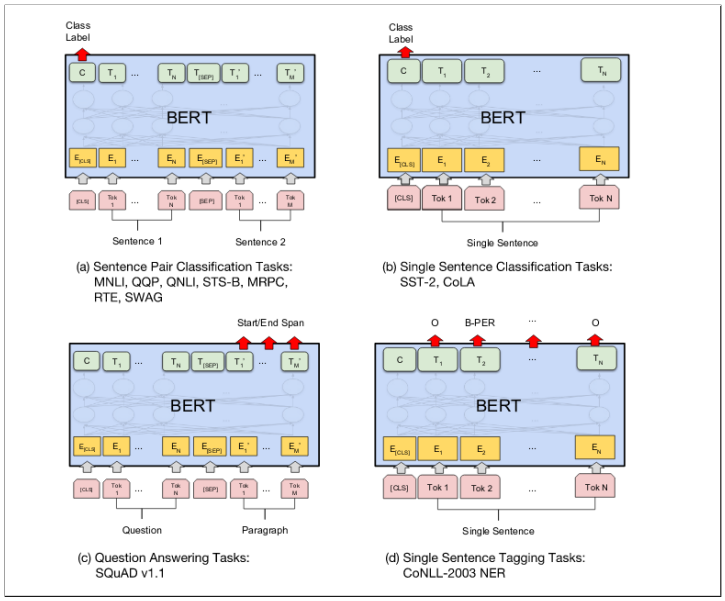


Figure 5.3.: The figure shows how BERT can be incorporated in various NLP tasks like sentence pair classification, single sentence classification, question answering and sentence tagging by adding only one additional output layer rather than training the whole model from scratch. [65]

- Training: After the raw data has been converted into features that can be fed to the BERT model, we shall load the BERT TF hub module to extract the computation graph. Next, we shall add a single new layer on top of the BERT model that will be trained to adapt BERT to the classification tasks. This process is also called fine-tuning of the BERT model. The model function is wrapped in a function that adapts the model to work for training, evaluation, and prediction. Checkpoints of the model are saved after a defined number of steps which contains the current updated parameter values of the model. The model is then evaluated using these checkpoints on the test data. [52]

- Inference: After the model has been trained, the saved checkpoints of the best model can be re-loaded to classify unseen text samples into the various target labels. The unseen samples can be fed to the model in the same way as was done with the training and validation samples using input examples and converting them into input features.

5.2.2. Image based datasets

In order to implement transfer learning on the image datasets, we shall use the pre-trained version of the Inception-Resnet-V2 model that has been trained upon the Imagenet dataset explained in 5.1.2 . The pre-trained model is then fine-tuned in two different ways, **Strategy I**: Firstly, by fine-tuning the last output layer only and freezing the earlier layers, **Strategy II**: Secondly, initializing the weights of the model with the weights of the pre-trained model trained on Imagenet and fine-tuning all the layers of the model. The images are loaded in the same way as explained earlier in 5.1.2 by using the torchvision data loader function to shuffle and load the train and validation image tensors in batches. The transfer learning process is implemented in two ways. The pre-trainedmodels package is used to load the model that has been trained previously on the Imagenet [66]. In Strategy I, the *requires_grad* variable for all the layers is set to False and a new fully connected neural_net output layer is added at the end of the model that is fine-tuned on our image datasets. This layer is the classification layer that predicts the probabilities of each of the class for a given image. In Strategy II, the *requires_grad* variable is omitted because of which each and every layer of the model is fine-tuned on our image datasets. In the same way, an additional fully connected layer is added to predict the classes for any given image. Cross Entropy loss is used as the loss function and SGD with momentum is used as the optimizer during training. Checkpoints of the model are saved after a defined number of steps which contains the current updated parameter values of the model. The model is periodically evaluated using these checkpoints on the validation data. The evaluation metrics are monitored and the training process is stopped once the selected metric, f1-score, converges indicating that the performance of the model cannot be improved further. After the model has been trained, the saved checkpoints of the best model are available for predictions. The checkpoints can be re-loaded to classify unseen image samples into the various target labels. The unseen samples can be fed to the model in the same way as was done with the training and validation samples using transforms and data loader.

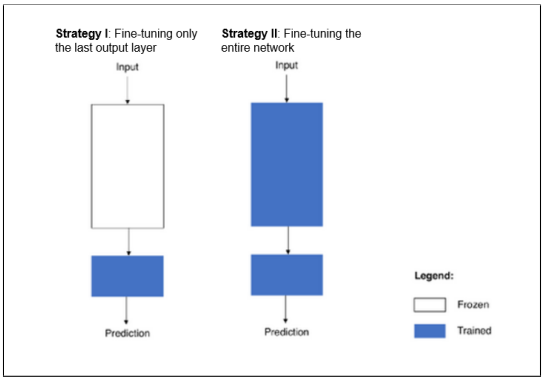


Figure 5.4.: Training strategies employed for image based transfer learning using the Inception.Resnet-V2 model

5.3. Multi-task Learning

One of the experiments fundamental to this thesis is the implementation of multi-task learning and to compare the model performance with another domain adaptation technique called transfer learning and also single task learning. Tensor2Tensor provides the option of multi-task learning training with the help of MultiProblem sub-classes [58]. In the section below, we shall discuss how we have tried to implement multi-task training on all 12 of our text based datasets in three different languages.

5.3.1. Text based datasets

- Data Generation: In the multi-task training implementation, we shall train on all the text based datasets across all task modalities and languages. We currently have twelve text based datasets in total namely, Category, Color, Brand and Gender dataset for English, German and French language. Along with the datasets that we have, we shall train a language modeling task with the Wikipedia corpora built using German, French, English and Romanian language with a vocabulary size of 64000 words. The language modeling dataset has approximately 3 billion samples. For the multi-task training, we shall use the language modeling vocabulary for all the 13 tasks that we shall train in parallel. In the data generation script, we shall firstly create a multi problem class that shall include all of the 13 sub-problems. In order to make all of the tasks compatible, we shall modify the vocabulary of each of our twelve tasks to use the vocabulary of the language model task. For each of the tasks, the data generation follows the same process as was in the single training data generation process in single task training explained in 5.1.1.

- Training: After the data generation phase, TFRecord files are created for each of the tasks along with the vocabulary file. Each of the task is associated with a task id which is implicitly assigned during the multi-problem data generation step. As the amount of data in each of the dataset varies, we shall use the threshold parameter for random sampling of each of the 13 datasets. The dataset with the highest sample size shall get a higher score as compared to one with a smaller sample size. For example, for two problems with weights 1 and 9 the first would be sampled 1/10 of the time and the other 9/10. [58] Each of the task uses a transformer model as was done in single task training in order to have a fair and valid comparison between single task and multi-task learning training. Checkpoints of the model are saved after a defined number of steps which contains the current updated parameter values of the model. The model is periodically evaluated using these checkpoints on the validation data. The evaluation metrics are monitored and the training process is stopped once the selected metrics converge indicating that the performance of the model cannot be improved further. The metrics are monitored easily using tensorboard.

- Inference: Once the tasks have been trained concurrently, unseen samples can be used in a file format for classification purposes. For prediction in multi-problems, the task

id needs to be specified in order to inform the model which task it should perform inference for. The task id that is assigned to each task we may want to use for inference can be found by instantiating the MultiProblem subclass and obtaining the value.

6. Experimental Setup and Results

Once the data is generate, it is now possible to conduct a set of experiments with respect to single task, transfer learning and multi-task learning. As part of the thesis, we shall conduct experiments using the three approaches and compare and evaluate the approaches to see which of them works well on our datasets. This shall help answer the three research questions that we have for the thesis 1.4. In this chapter, we shall firstly discuss the experimental setup that is required to conduct the experiments followed by outlining the results achieved from them.

6.1. Experimental Setup

6.1.1. Hardware

As discussed earlier, deep learning requires special hardware for training. Therefore for our experiments, we shall use Graphical Processing Units to train our models. Training of each of the model for the various tasks is time consuming and requires a lot of memory [67]. The GPUs help in tackling this problem. For the experiments, we use a number of deep learning frameworks and libraries like T2T, pytorch and tensorflow. All of these packages are capable of using available GPUs to expedite the training process which would be rather impossible to perform if local machines were used. The training can also be distributed across multiple GPUs using T2T for text based datasets and using Apex and Pytorch using image based datasets. Apex is a package which enables mixed precision and distributed training using Pytorch [68]. Apex enables Tensor Core-accelerated training in only a few lines of code. For our image based experiments, we have used FP16 mixed precision. This option casts enables dynamic loss scaling. The table below shows the specifications of the machine we used to train the models for each of our tasks.

	Machine
Name	DGX-1
GPUs	8x Tesla V100
Core	41k
Memory	8x 16GB

Table 6.1.: Table depicting the details of the GPU used for the experiments.

6.1.2. Hyperparameters

In the section below we shall discuss the hyperparameters that have been used during each of the training process. The hyperparameter choices have been made by studying research papers that have implemented state-of-the-art techniques. This is a vital research question that we have tried to solved during the tenure of the thesis. As we have used different architectures and hyperparameter combinations, we shall list them down separately for text based and image based experiments in the figures below.

1. Text based Datasets:

	Single Task	Transfer learning	Multi-task learning
Model	Transformer	BERT	Tranformer
Hidden size	512	768	1024
Filter size	2048	3072	8192
Batch size	4096	32	1024
Optimizer	Adam	Adam	Adam
Maximum sequence length	300	300	512
GPUs used	1	1	8

Table 6.2.: Hyperparameters used for the text based experiments.

2. Image based Datasets:

	Single Task	Transfer learning: Strategy I	Transfer Learning Strategy II
Model	Inception-Resnet-v2	Inception-Resnet-v2	Inception-Resnet-v2
Fine tuning	No	Last Layer only	Whole Network
Pre-Trained	No	Yes	Yes
Batch size	196	1024	196
Optimizer	Adam	Adam	Adam
GPUs used	7	1	7

Table 6.3.: Hyperparameters used for the image based experiments.

6.1.3. Software

- Tensor2Tensor: Tensor2Tensor or T2T is a library of deep learning models and datasets developed by the Google Brain team. It uses tensorflow and enables fast prototyping of customized deep learning problems and integration of new datasets. The library consists of numerous baseline models which can be trained on new datasets using a variety of different hyperparameter sets. The performance of these architectures can

then be compared and the best model can be chosen among the various options. The existing models and hyperparameters can also be customized according to the needs of the problem. The Transformer is one of the many models that has been implemented through the T2T library provides and is the one we shall use to conduct single task experiments on text based datasets. Using T2T, we shall integrate our datasets using data-generators for our text classification tasks across three different languages. [58]

- pre-trainedmodels: The pre-trainedmodels package is very similar to the torchvision models sub-package that contains definitions of state-of-the-art models that can be used to reproduce results for image related tasks such as classification, segmentation and object detection. [57]

  For single task training, the non-pre-trained version of the Inception-Resnet-V2 model is downloaded. The weights of the model are re-initialized and the model is trained from scratch for the classification tasks. In case of the transfer learning experiments, the pre-trained model is downloaded and the model is fine-tuned using two different strategies mentioned earlier. [66]

6.1.4. Evaluation Metrics

In this section we shall discuss the metrics that we have used during the experiments to evaluate the training and validation process of the classification tasks. Since all the tasks that we have are text and image classification tasks, we shall use the metrics that are commonly used to evaluate a text classification model [69]. Additionally, the multi-task learning training involves a language modeling task along with our classification tasks. In the section below, we shall list down all the metrics that we have used in our experiments.

Classification:

- Accuracy: Accuracy is a metric which is used to measure the performance of a classification model. It is defined as the fraction of predictions that were accurate. It is formally defined as :

$$Accuracy = \frac{Number of Correct Predictions}{Total Number of Predictions}$$

- Precision: In case of multi-class classification tasks as is in our case, accuracy is not sufficient to assess a model. This is primarily important for datasets where the classes are not balanced. Accuracy is a metric that only takes the correct predictions into consideration whereas precision is a metric that takes into consideration how precise a model is at predictions. It is basically the fraction of true positives to the sum of true positives and false positives. This takes into consideration how many false positives do we have from the model. Therefore, a model with high accuracy will not necessarily have high precision. It is formally defined as:

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives}$$

For our experiments, precision is calculated for multiple classes using the macro-averaging method. The precision for each of the class is summed up and averaged out.

- Recall: In scenarios where there is a high cost involved in getting low number of false negatives, the metric that could be used is recall. It takes into consideration how the model is in making wrong predictions. It is basically the fraction of true positives to the sum of true positives and false negatives. This takes into consideration how many false negatives we have from the model. Therefore, a model with high accuracy will not necessarily have high recall. It is formally defined as:

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives}$$

In the same way, recall is calculated for multiple classes using the macro-averaging method. The recall for each of the class is summed up and averaged out.

- F1-Score: F1-Score is a metric that is appropriate for evaluating classification models for uneven class distribution as it takes into consideration both precision and recall. [70]

$$F1 = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

The macro-average F1-Score is simply the harmonic mean of these macro-averaged precision and macro-averaged recall.

6.2. Experimental Results

In this section we shall discuss the results we have achieved by conducting each of the experiments using single task, transfer learning and multi-task learning. As all of the tasks are classification tasks, each of the results are presented using two metrics namely, accuracy and f1-score. The text dataset is divided into three languages namely, English, German and French and each of the language is further divided into four tasks namely, Category, Color, Brand and Gender hence, resulting in 12 tasks whereas the image dataset is divided into three countries namely, English, German and Italian and each of the language is further divided into the same four kind of tasks.

6.2.1. Single Task Learning

1. Text based datasets: In order to create a baseline comparable to the models based upon domain adaptation techniques, we train a model based upon Transformer Base architecture for single tasks on 12 different text based datasets. Each of the task is trained independently on each of the datasets resulting in 12 different tasks. In this way, we were able to evaluate the single task results and compare it with the results achieved from the transfer learning and multi-task learning training done using the same 12 text based datasets. Each of the 12 task was trained for 250k training steps using the Transformer base model. The results achieved for each of the three languages for each of the task type is pretty similar. The average accuracy and f1-score for the category prediction task for all the three languages is 77% and 57% respectively with DE having the highest accuracy and f1-score of 79 and 59 respectively. For the color prediction task, the average accuracy and f1-score for all the three languages is 39% and 21% respectively which is much lower than that for category. DE again has the best performance out of the three languages. In case of brand prediction, the average accuracy is 88% and average f1-score is 73%. The average accuracy and f1-score for the gender task is the highest with values of 92% and 84% respectively. The results for DE and UK have been visualized in figure 6.1 and 6.2.

2. Image based datasets: Through the experiments we not only want to create a comparison between single task and domain adaptation techniques but also create a comparison between experiments using text based datasets and image based datasets. Therefore, we create a baseline comparable to domain adaptation techniques using the images datasets for the 12 tasks mentioned previously. We train a model based upon the non-pre-trained version of the Inception-Resnet-v2 architecture. The image classification tasks are also evaluated using the same metrics namely, accuracy and f1-score. The models were trained until the f1-score converged and the model could not be improved further. The average accuracy and f1-score for the category prediction task for all the three languages is 85% and 85% respectively with IT having the highest accuracy and f1-score of 86 and 84 respectively. For the color prediction task, the average accuracy and f1-score for all the three languages is 69% and 74% respectively which is much lower than that for category. DE has the best performance out of the three languages. In case of brand prediction, the average accuracy is the worst with values of 54% and 52% for average accuracy and average f1-score. The average accuracy and f1-score for the gender task is the highest with values of 95% and 95% respectively. The results for DE and UK have been visualized in figure 6.1 and 6.2.

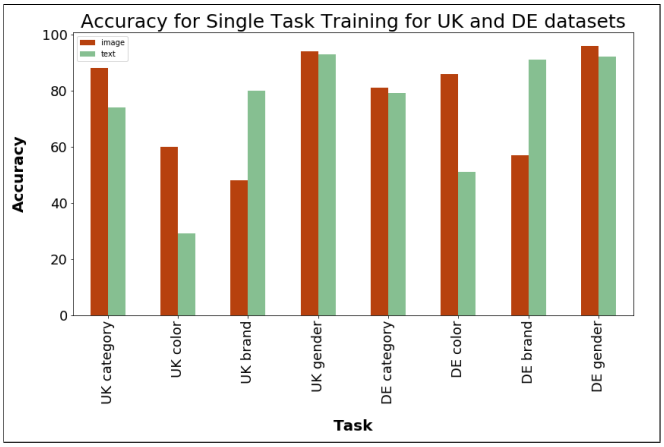


Figure 6.1.: Accuracy scores for STL for the DE and UK tasks for both image and text based datasets

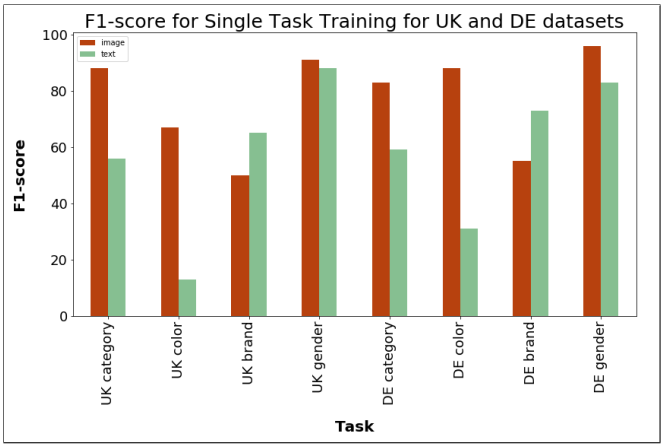


Figure 6.2.: F1 scores for STL for the DE and UK tasks for both image and text based datasets

6.2.2. Transfer Learning

1. Text based datasets: In order to conduct experiments using the transfer learning approach, we train a model based upon the base version of the BERT model. We train the model on each of the 12 text based datasets independently resulting in 12 different tasks. In this way, we were able to evaluate the BERT model and compare it with the results achieved from the single task learning and multi-task learning training done using the same 12 text based datasets. As all of the tasks are classification tasks, each of the results are presented using two metrics namely, accuracy and f1-score. Each of the 12 task was trained for 5 epochs using the BERT base model. The results achieved for each of the three languages for each of the task type is pretty similar. The average accuracy and f1-score for the category prediction task for all the three languages is 98% and 97% respectively with UK having the highest accuracy and f1-score of 98 and 98 respectively. For the color prediction task, the average accuracy and f1-score for all the three languages is 60% and 53% respectively which is way lower than that for category. DE has the best performance out of the three languages with accuracy of 69% and f1-score of 58%. In case of brand prediction, the average accuracy is 100% for both average accuracy and average f1-score. The average accuracy and f1-score for the gender task is the second highest with values of 100% and 99% respectively. The results have been visualized in figure 6.5 and 6.6.

2. Image based datasets: In order to apply transfer learning on the image based datasets, we shall use the Inception-Resnet-v2 model pre-trained on the Imagenet dataset in two ways as explained in 5.2.2. Firstly, we shall freeze all of the layers of the model and re-train only the last output layer of the model. This method will be referred as Strategy I transfer learning going forward. Secondly, we shall use the initialized weights of the model and re-train all the layers including the last output layer. We shall refer to this approach as Strategy II transfer learning. Each of the model type was trained until the f1-score converged and the model could not be improved further.

   The average accuracy and f1-score for the category prediction task for all the three languages for Strategy I is 56% and 56% respectively and for Strategy II is 81% and 80% . UK has the highest accuracy and f1-score of 88 and 86 respectively. For the color prediction task for all the three languages for Strategy I average accuracy is 43% and 43% respectively and for Strategy II is 77% and 80% respectively. In case of brand prediction, the average accuracy is the worst with values of 45% and 43% for average accuracy and average f1-score for Strategy-I and 57% and 59%. The average accuracy and f1-score for the gender task is good for both Strategy I Strategy II with values of 93% and 91% respectively for Strategy I and 95% and 95% for Strategy II. The results for DE and UK have been visualized in figure 6.3 , 6.4 , 6.5 and 6.6.

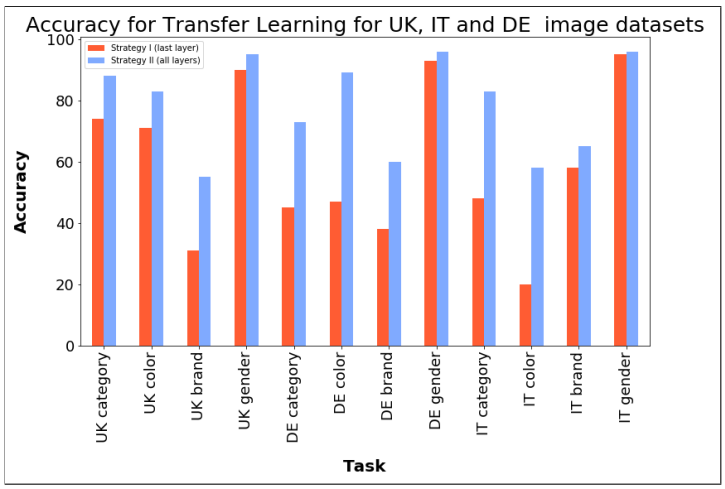


Figure 6.3.: Accuracy scores for the two strategies used to implement TL on the image datasets.

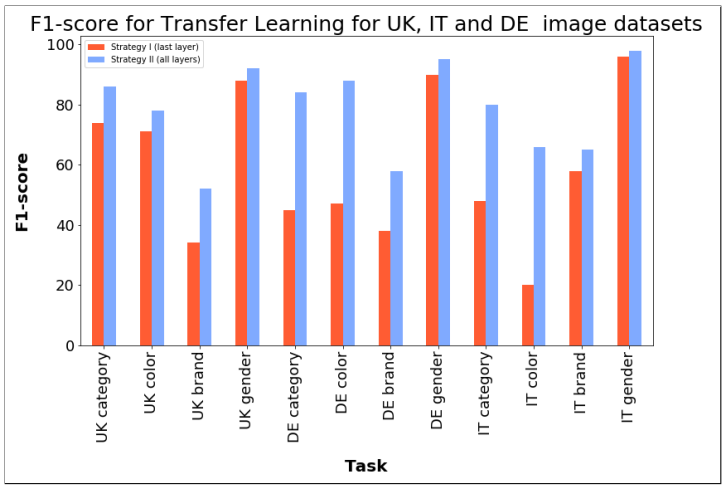


Figure 6.4.: F1 scores for the two strategies used to implement TL on the image datasets.

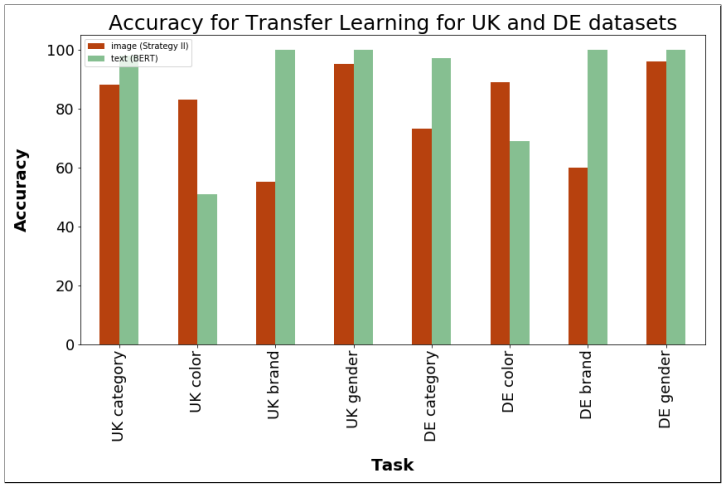


Figure 6.5.: Accuracy scores for TL for the DE and UK tasks for both image and text based datasets.

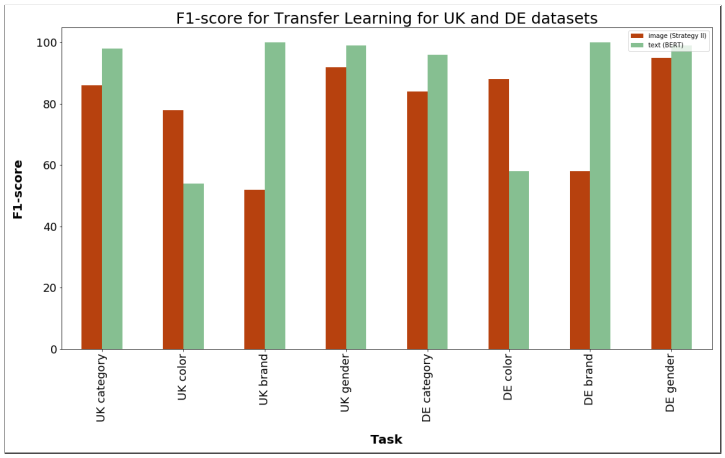


Figure 6.6.: F1 scores for TL for the DE and UK tasks for both image and text based datasets.

6.2.3. Multi-task Learning

1. Text based datasets: We conducted experiment to implement multi-task learning and to compare and see if it performs better than transfer learning. For the experiment, we trained all of the 12 tasks concurrently along with a language modeling task using the Transformer base model for each of the task. Only accuracy was used to measure the performance of the model because T2T only provides multi-task learning feature in a beta version. This version currently does not support incorporating additional metrics for classification tasks for multi-task training. Therefore, we were unable to calculate f1-score for the tasks. Therefore, we were able to compare multi-task model with single task and transfer learning for text based datasets in terms of accuracy. The accuracy achieved through the multi-task experiments for each of the task is better than that achieved from single task learning. However, in comparison with transfer learning, the results are very close in terms of accuracy. The average accuracy for the category prediction task for all the three languages is 97% respectively with UK having the highest accuracy of 98% . For the color prediction task, the average accuracy for all the three languages is 62% respectively which is way lower than that for category. DE has the best performance out of the three languages with accuracy of 64% . In case of brand prediction, the average accuracy is 100% for average accuracy. The average accuracy for the gender task is the second highest with a value of 97%. The results have been visualized in figure 6.7.

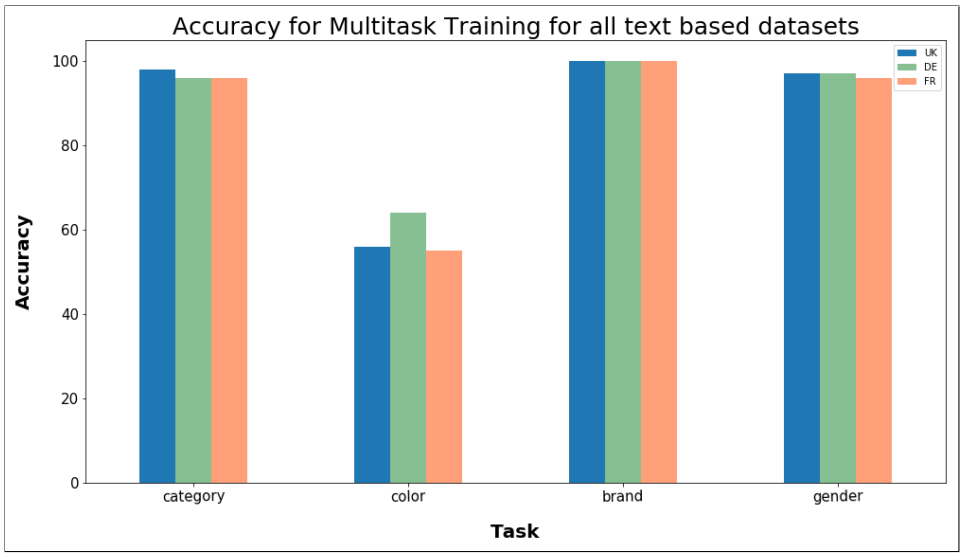


Figure 6.7.: Accuracy for all the text based datasets for Multi-task training.

7. Discussion

7.1. Overview

In this chapter we shall compare the results of the different experiments and see which approach works best for the problems we set out to solve at the start of the thesis. Below, we provide with two detailed tables containing results for each of the experiment conducted for both text and image based problems. Following this, we provide a comparison across four major areas namely, languages, task types, modalities and approaches. The details of each has been provided in the sub-sections below.

Modality	Dataset	Single Task		Transfer Learning		Multi-task learning	
		Accuracy	F1-score	Accuracy	F1-score	Accuracy	F1-score
Text	Uk Category	74	56	98	98	98	-
	UK Color	29	13	51	54	56	-
	UK Brand	80	65	100	100	100	-
	UK Gender	93	88	100	99	97	-
	DE Category	79	59	98	96	96	-
	DE Color	51	31	69	58	64	-
	DE Brand	92	73	100	100	100	-
	DE Gender	92	83	100	99	97	-
	FR Category	78	56	98	97	96	-
	FR Color	36	18	60	48	55	-
	FR Brand	93	81	100	100	100	-
	FR Gender	90	81	100	100	96	-

Table 7.1.: Table containing the Accuracy and F1-scores for all the text based models.

Modality	Dataset	Single Task		TL - Strategy I		TL - Strategy II	
		Accuracy	F1-score	Accuracy	F1-score	Accuracy	F1-score
Image	Uk Category	88	88	74	74	88	86
	UK Color	60	67	71	71	83	78
	UK Brand	48	50	31	34	55	52
	UK Gender	94	91	90	88	95	92
	DE Category	81	83	45	45	73	84
	DE Color	86	88	47	47	89	88
	DE Brand	57	55	38	38	60	58
	DE Gender	96	96	93	90	96	95
	IT Category	86	84	48	48	83	80
	IT Color	60	67	20	20	58	65
	IT Brand	56	53	65	58	55	66
	IT Gender	96	97	96	96	96	98

Table 7.2.: Table containing the Accuracy and F1-scores for all the image based models.

7.1.1. Across Task Languages

When we compare the performance of the experiments across all the text based datasets for English, French and German language, we notice a common trend for all the tasks across all methodologies. For the Gender and Brand prediction tasks, we cannot conclude from the results that one language is superior than the rest. The experiments are not favored by using any particular language dataset for the two tasks as they have similar performance scores across all three languages. Only in case of single task training for Brand prediction, UK dataset does not perform as good as the FR and DE datasets. The Color prediction task has highest performance for the DE dataset followed by FR and UK in case of the BERT model. The same trend is observed for the Color task in case of single task learning. In case of multi-task learning, DE stays the best followed by UK and then FR. Therefore, it is safe to say for all our experiments for the Color prediction task, the German language seems to have performed slightly better in comparison with the other two languages. In case of the Category task, the languages have different performance results for the different approaches. For single task, DE performs best followed by FR and UK whereas in case of the BERT model, the performances are very close for all. In case of multi-task learning as well, performances are comparable with UK on top followed by DE and FR having the same performance scores. A detailed visualization of the performances across task languages is visualized below.

7.1.2. Across Task Types

Across task types i.e. Brand, Gender, Color and Category, the trend is similar for all the approaches when it comes to the text based datasets. The highest performing task in terms

of both accuracy and f1-score is the Brand prediction task. On manual inspection, it was noticed that majority of the samples have the brand name in the beginning of the description. Thus, the models are easily able to identify this and make accurate predictions. The Gender prediction task comes second after Brand for all the approaches. This is primarily because this task does not have hundreds of unique labels as is the case for the rest of the tasks. Each language has as few as six to eight unique labels only for this task hence, making it easier for the models to perform well. Next in line is the Category prediction task. The worst performing task for text based datasets is the Color prediction task. On inspecting text samples, it was observed that generally the descriptions do not contain information regarding the color of the product and maybe because of this the text based models do not perform as well. In case of the image based datasets, the Gender prediction task performs best. This is mostly because the images of the products also have the models sporting the products to indicate the target gender of the product. This is followed by the Category task and the Color task. The worst performing task for images is the Brand prediction task. This is intuitive as the images do not contain features through which one can identify the brand of a given product. The figures below visualize the performance scores across all tasks in figure 7.1 and figure 7.2.

7.1.3. Across Task Modalities

Across task modalities i.e. image based and text based experiments, we see different performance scores for different kinds of tasks. In case of the color prediction tasks, image based tasks perform better than text based tasks. This is intuitive because the model is able to learn better features pertaining to color from images as compared to product descriptions which may or may not contain color related information. It is interesting to see how single task image based training performs better than both transfer learning and multi-task learning for text based tasks indicating that images are superior when it comes to predicting color. However, performance from the first approach of image based transfer learning i.e. by re-training only last layer is pretty low in comparison with re-training all layers or text based transfer learning using the BERT model. In terms of the other tasks, text based transfer learning performs much better as compared to image based transfer learning with re-training all layers. We did not perform multi-task learning on the image based datasets. Therefore we cannot compare multi-task learning between image and text based datasets.

7.1.4. Across Task Approaches

One of the research questions for the thesis was to compare multi-task learning, transfer learning and single task learning. Through the 25 experiments conducted on text based datasets and 36 experiments on image based datasets, we conclude that domain adaptation techniques perform better than single task training for text based datasets. However, in case of images, transfer learning by fine-tuning all layers and single task learning are pretty close in terms of performance. Single task training performs well in case of image based datasets because the dataset is not extremely scarce with at least a few hundred images per

class. This enables the model to learn and perform well. If we compare the two domain adaptation techniques then we can see that the performance in terms of both accuracy is very close. For some tasks and languages, multi-task learning has a little higher accuracy whereas for some, transfer learning performs better with higher accuracy. As we were unable to calculate f1-scores for multi-task for the entire validation dataset, we cannot compare the two approaches with respect to this metric. The figures below will give a more detailed picture of the performances achieved from each of the approach. It is however safe to say that domain adaptation techniques work well in case of classification tasks for both text and image based datasets.

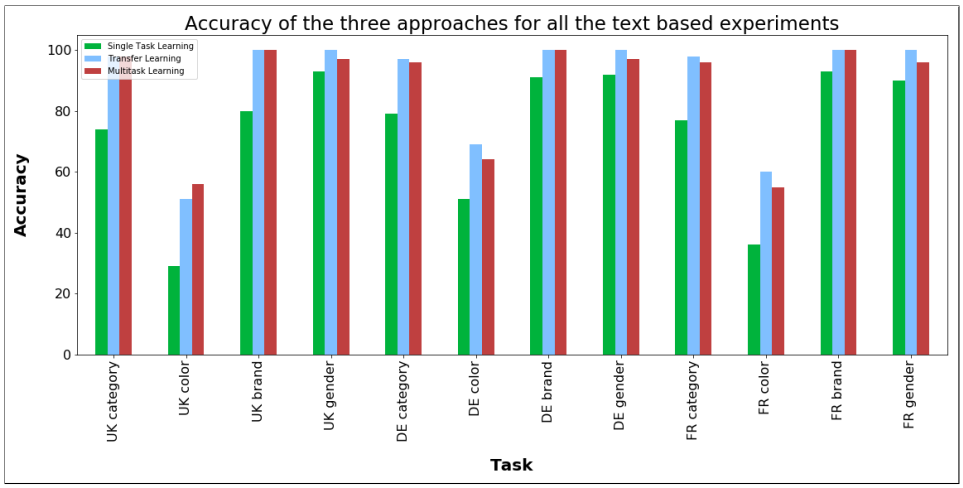


Figure 7.1.: Accuracy for all the approaches for the text based datasets.

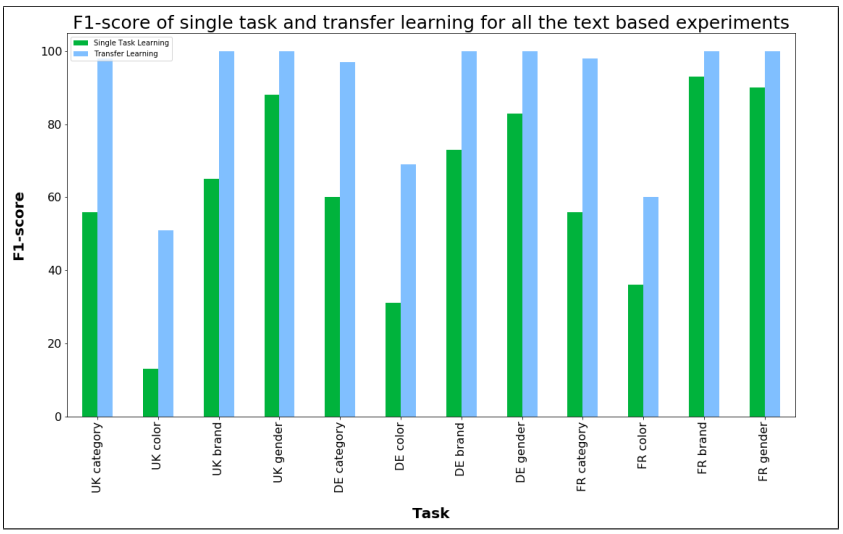


Figure 7.2.: F1-scores for STL and TL for the text based datasets.

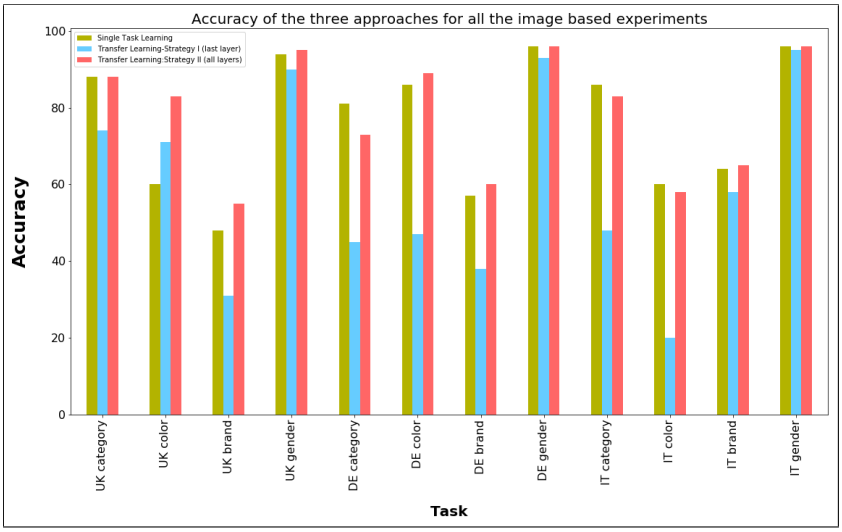


Figure 7.3.: Accuracy for all the approaches for the image based datasets.

Figure 7.4.: F1-scores for all the approaches for the image based datasets.

7.2. Survey

In order to answer the third research question which was to see if at all predictions of missing product information helps user-experience on e-commerce platforms 1.4, we conducted an anonymous survey after the completion of all the experiments targeting users that generally buy products online. The survey was taken by 79 individuals, 57% of whom were males while the rest females. The respondents were majorly from Germany, India, United States of America and Singapore. Out of the 74 respondents, 57% had bought an item online within the last week of taking the survey whereas 24% had bought an item within the last month of taking the survey. The survey presented 8 questions out of which 4 were image based questions and 4 were text based. The first 4 questions provided with two images of replicated dummy versions of original product detail pages of items containing features predicted by our models, the first image had a predicted product feature (color, brand, gender, category) missing from the detail page text whereas the second option included the predicted feature. The respondents were asked to choose which detail page they would prefer to see while making a buying decision with respect to a given product. In case of brand, 92% of the people would prefer a detail page which includes the brand name in the description text whereas in case of gender, 75% of the people would prefer a detail page which includes the target gender of the product for example, (*for women, for men, for kids, unisex etc.*). 75% of the people prefer the color of the product to be present in the description text/title. Indicating that the image of the item is not sufficient to decipher its color. The last of the 4 image based question consisted of two detail pages, one which included multiple but related category names in the product title while the second included only specific category name. 71% of the people preferred the longer title with numerous category names.

The next 4 questions were text based in which users were asked if color, category, brand and target gender of the product is important in the detail page text for them to make a buying decision. The two most important features were brand name of the product and the target gender of the product with approximately 82% of the people checking this information before making a buying decision. This was followed by the color of the product with 74% of the people looking at the color of the product in the text before making a decision. Only 52% of the respondents look at the category of the product in the detail page before deciding to buy or not to buy the product in question. This could be due to the fact that the image of the product is sufficient to decipher the category of the product.

Through the survey, we can conclude that majority of the people depend upon the four predicted features of the product to be included in the description text for making buying decisions on e-commerce platforms with brand and gender having the highest importance followed by color and category of the product. We can therefore conclude that predicting missing product features is an important problem in this domain and employing domain adaptation techniques in predicting such missing information is therefore useful in the e-commerce domain to enhance user-experience.

8. Conclusion

Through this thesis, we compiled an e-commerce product dataset from scratch by scraping regional websites for an e-commerce giant. The datasets offer as a starting point for conducting various experiments using deep learning techniques in the field of natural language processing and computer vision. The datasets consist of product details like description, bullet points, specifications, brand, gender, color, category and image in three languages namely English, Germany and French. The images have been scraped from the German, English and Italian website. The data generated from scratch has been used for various text and image classification tasks using various approaches. Considering that the dataset dos not have balanced classes and there is limited data in the range of a couple of hundred thousands rather than millions or billions, we used techniques in domain adaptation that helps in the data shortage problem. We also tried to see if these approaches are better than training models from scratch without involving any kind of knowledge transfer or sharing. We tried to identify ideal scenarios for the experiments by identifying architectures and hyperparameters for each of the classification task. We then compared the results to see if using such techniques is beneficial in the e-commerce domain in order to improve product catalog quality and predict missing information.

In order to implement single task training for images, we used a non-pre-trained version of the Inception-Resnet-v2 model and compared it with the same model trained on the Imagenet dataset and fine-tuned on our customized dataset. In case of the text classification tasks, we implemented Single Task training using the Transformer base model via Tensor2Tensor package and compared it with transfer learning using the BERT model and 12 Transformer models and Language models trained concurrently through multi-task learning.

After the implementation, we compared the results across the three approaches (single task, transfer learning and multi-task learning), two modalities (image and text), three languages (German, English and French) and four (Category, Color, Brand and Gender) tasks. The comparison results in some straightforward conclusions. On comparing the three approaches, domain adaptation techniques i.e. Transfer learning and multi-task learning perform much better as compared to Single Task for text based datasets in terms of both accuracy and f1-scores. If we just compare multi-task learning and transfer learning then the comparison has varied results depending upon the language and the task. However, the results are very close in terms of accuracy and it is safe to say that both approaches are good options to choose from. Multi-task learning takes much longer however, if we compare its training with transfer learning using BERT.

For the image classifications, Strategy I of transfer learning which involves fine-tuning only the last layer performs the worst and Strategy II and Single task using Inception-Resnet-V2 both perform the best. The performance scores are pretty close for both of these approaches.

It is interesting to see that single task learning performs well for the images. This could probably be because the dataset is not extremely small in size and has at least a few hundred samples for each label enabling the model to learn and converge with good accuracy and f1-score.

If we compare the the text based results with the image based results, we observe that the results are at par for the gender and category task. However, for color, images perform way better when compared to text. This is because the features extracted from images is better at color prediction as compared to extracting features using text. It is right the opposite for brand with text having a higher accuracy as compared to images. This is intuitive as images do not contain any information regarding the brand of the product. On comparing the three languages, we see a similar trend in the performance with respect to each of the task. While gender and brand prediction has the best scores followed by category, color prediction has the worst scores for all the languages when it comes to text. In case of images, gender has the best scores followed by category, color and brand.

Therefore, through the thesis work, we were able to answer the research questions we had aimed at solving through various deep learning experiments and an online survey. In case of limited data, unbalanced classes and a need for fast training and good results, domain adaptation techniques like transfer learning and multi-task learning are a clear winner in comparison to single task learning for text based datasets. If we have sufficient images, we could try both transfer learning and Training from scratch using a sophisticated architecture such as the Inception-Resnet-v2. In case of transfer learning, it is probably better to retrain all the layers than retraining only the output layer. This highly depends upon the similarity of the target dataset to the source dataset, Imagenet in our case. In case of physical features like color, curves, handles, wheels etc. images work better than texts. However, if we want to predict labels like brand, gender etc, we could use text which could contain textual information about the products like in our case the title of the product contains the name of the brand the product belongs to.

We showed that multi-task learning and transfer learning can be beneficial in the e-commerce domain to predict missing product information and hence improve user-experience through the conducted survey. In the same way, we could predict other features of the product like the size, weight, material etc which could also add organizational benefits like stock management, efficiency in product delivery and a high quality product catalog. The techniques can be expanded to solve other problems in the e-commerce domain like generating high quality product titles, product summary etc.

9. Future Research

Through the thesis work, we have generated datasets that can be further used for performing various tasks in the field of natural language processing and computer vision. The corpora can be used for tasks such as machine translation, text summarization and further text classifications. Another use-case suitable for using the datasets in the e-commerce domain is to automatically generate the title of a given product given its description and image. Machine translation could be particularly useful as we have corpora from three different languages and an auto-translate feature for third party sellers that list products on such websites is a valuable feature. The dataset could also be used to build recommender systems which could be content based or collaborative in nature. Recommender systems are a very important feature that are being used on e-commerce platforms to enable users to make informed decisions while browsing such websites.

In terms of the techniques applied already, we could expand the work further by tuning different combinations of hyperparameters. Due to the limited time for the thesis, we were unable to try out the various multitudes of hyperparameter combinations. In case of transfer learning for images, an interesting area of research could be identifying the ideal layer till which we should freeze a pre-trained network. Different combinations could be tried to see what works best for the datasets. In terms of multi-task learning, different combinations of tasks could be trained together to see which group performs best together. Auxiliary tasks could also be added to see if it improves or hurts performance. Due to the limited time in hand for the thesis, we were unable to implement multi-task learning for images. This could also be another interesting area of research. Conducting experiments with both text and image as input to a model would also be another area that could be interesting to work in. This also creates a great potential for ensemble models that combine various models together and result in a boost of performance. In conclusion, we have implemented deep learning techniques in the e-commerce domain through this thesis and hope to provide a starting point for further research in the field.

A. Appendix

A.1. Figures

As we were unable to fit the entire input text for all the text data samples in 4.1, we present the detailed version of it below:

Dataset	Input Sample	Label
FR Category	7 For All Mankind - Jeans - Bootcut - Femme. 98% Coton, 2% Élasthanne . Lavage en machine, 30 max. Fermeture: Fermeture éclair. Taille normale.	Jeans
FR Color	Bloch Criss Cross, Chaussures de Danse Moderne and Jazz Fille. Basket de dance à semelle partagée avec soutien de voûte intégré. Semelles basses sans frottement, profilées et flexibles avec point de rotation. Tige en maille respirante et doublure en dri-lex. Dessus: Synthétique. Doublure: Mesh. Semelle intérieure: Textile. Matériau de semelle: Synthétique. Type de talons: Plat. Hauteur de talons: 0.25 pouces. Fermeture: Lacets.	Noir
FR Brand	adidas FEF H JSY T-Shirt pour Homme Rouge/Or. Dominez le terrain en portant le maillot adidas Spanish Football Federation Home pour homme. Réplique du maillot porté par La Furia Roja à domicile, ce maillot est conçu avec CLIMACOOL, qui régule la transpiration, et le badge authentique de la Fédération d'Espagne de football à gauche sur la poitrine. CLIMACOOL assure une régulation optimale de la transpiration. Badge authentique de la Fédération d'Espagne de football à gauche sur la poitrine. Coupe régulière Interlock 100 % polyester. Fabricant: Adidas. Matériau: 100% Polyester Adidas Climacool Technologie. Référence d'article X10937	Adidas
FR Gender	Chic Feet , Sandales pour femme. Numéro du modèle de l'article: Chic Feet 77. ASIN: B00CJRERZ6. Date de mise en ligne sur Amazon.fr : 13 mars 2014	Femme

Table A.1.: Detailed samples from the FR dataset.

Dataset	Input Sample	Label
UK Category	Holstyle 0.6cm Heel Lift Half Insoles for Loafer, Sneakers, Dress Shoes Mesh Black: Amazon.co.uk: Shoes and Bags Your loafes or slip-on shoes has no cushion? If so, Holstyle synthetic leather heel pad or half shoe insoles is perfect for your shoes. Ideal for who works standing or have to wear dress shoes without cushion all day. Urethane, Well Known High Quality Shoe Material.,Ideal for people having leg length discrepancy or work standing all day with hard shoes.,Add soft heel cushion on your hard, thin or flat soles of shoes.,Luxury Synthetic Leather upper layer,One Size Fits All. Product Dimensions:15 x 9 x 0.6 cm ; 49.9 g , Boxed-product Weight: 113 g , Delivery Destinations: Visit the Delivery Destinations Help page to see where this item can be delivered.Find out more about our Delivery Rates and Returns Policy , Manufacturer reference: halfloafermesh , ASIN: B00LVMO93U , Date first available at Amazon.co.uk: 17 Jun. 2015 , Average Customer Review: Be the first to review this item , Amazon Bestsellers Rank:192,704 in Shoes and Bags (See Top 100 in Shoes and Bags).	Insoles-Comfort
UK Color	Mini dress with deep V-neck - Pink - 14-16: Amazon.co.uk: Clothing. Are you looking for a sexy dress for the new year party? Our Mini Dress with Deep V Cut is a perfect choice. The V cut shows the beautiful Dekollette. The zip?Closure on the neck style make the dress even seductive. The Mini length betonnt the long legs. The dress is with boots or dress footwear with killer heels look. Makes it easy to slide out the sexy dress the new year party sea. In Mini length, Betonnt these attractive legs,Minimal design only 20Â x 10Â x 8Â mm, refined and classic,Cut-Out,Slim Fit,Long Sleeve,Original collar cut.	Pink
UK Brand	Nike Slam Women's Dri-Fit Tennis Skirt - Black-XL: Amazon.co.uk: Clothing.	Nike
UK Gender	Cinda Baby Girls Christening Party Dress with Shoes: Amazon.co.uk: Clothing. Swirling flowers on the bodice,Material flower centre of waistline,Polyester,Three layer skirt,zip back and tie back sash,rose pattern shoes.	Baby girl

Table A.2.: Detailed samples from the UK dataset.

Dataset	Input Sample	Label
DE Category	Rockabella Ivy Kleid schwarz/weiß: Amazon.de: Bekleidung. Swing,Modellnummer: D-IVY-BW-1.	Kleid
DE Color	ESSEX GLAM - Damen Riemchen Plateau Sandalen Stiletto Absatz: Amazon.de: Schuhe and Handtaschen. Essex Glam - Damen Riemchen Plateau Sandalen Stiletto Absatz. Obermaterial: Synthetik,Innenmaterial: Synthetik,Sohle: Gummi,Verschluss: Schnürsenkel,Absatzhöhe: 1 cm,Absatzform: Flach,Materialzusammensetzung: Obermaterial: Textil / sonstige MaterialÂ /Futter und Decksohle: Textil /Laufsohle: Gummi,Schuhweite: normal.	Schwarz
DE Brand	Wrangler Herren Jeansjacke Auth Western: Amazon.de: Bekleidung. Klassische Western-Style Jacket,Klassische Western-Style Jacket,Kragenform: Button-down,Langarm,Normaler Bund,Verschluss: Knopfleiste,100% Baumwolle,Blouson,Pflegehinweis: Maschinenwäsche kalt (30 max),Modellnummer: W41001705,Klassische Western-Style Jacket,Klassische Western-Style Jacket,Klassische Western-Style Jacket	Wrangler
DE Gender	Sakkas Azalea Stein gewaschen gestickte Kunstseide Korsett Stil Kleid: Amazon.de: Bekleidung. Features schöne Stickerei, breitstreifen, Korsett Stil vorne und verklemmte zurück mit hinteren Krawatte. Einzigartige Design-Features reiche feste Farben, Figur schmeichelnde Form, zarte gestickte abgestufte Rock-Panels und eine schiere Crepe Saum. Handwäsche separat in kaltem Wasser. Linie trocken Importiert. Aufrechtzuerhalten Material: 100% Rayon Ãœber Sakkas Store: Sakkas bietet trendige Designer inspirierte Mode bei tiefen Rabatten! Wir arbeiten Tag und Nacht, um Ihnen qualitativ hochwertige Kleidung und Accessoires für einen Bruchteil des Preises zu bringen, den Sie an den Kaufhausern zahlen. Unsere unglaublichen Angebote verkaufen schnell, also warten Sie nicht! S / M ((Passend für ca. US Kleid Große 0-12, UK Große 6-16, EU-Große 34-44) Max Oberweite von 40 Zoll, L / XL (US-Kleid-Große 10-18, UK Große 14 bis 22, EU-Große 42-50) Max Oberweite von 45 Zoll, 1X / 2X (US KleiderGroße 12-2X, UK Große 16 bis 24, EU-Große 44-52) Max Oberweite von 48 Zoll,Ungefähre Länge = 50 Zoll (128 cm) MaÃŸ Schulter bis zum Saum,Ã„rmellos,100% Viskose,A-Linie,Verfügt schönen Stickereien, breiten Trägern, Korsett Stil Front und gesmokt Röcken mit hinten Binde.,Einzigartiges Design verfügt reichen Uni-Farben, figurschmeichelnder Form, zart bestickte Stufenrock-Platten und eine schier Krepp Saum.,Hand waschen separat in kaltem Wasser. Leine trocknen. Importiert. | Material: 100% Rayon	Damen

Table A.3.: Detailed samples from the DE dataset.

List of Figures

1.1. A figure depicting the stages of the thesis work. 6
1.2. A gantt chart showing the timeline of the thesis. 7

2.1. Architecture of LeNet-5 used for digit classification. 10
2.2. Sequential LSTM blocks. 10

4.1. Image samples for the 4 task types in the DE dataset. 30
4.2. Distribution of the input length size for the 4 task types in the UK text datasets. 34
4.3. Distribution of the Input length size for the 4 task types in the DE text datasets. 35
4.4. Distribution of the Input length size for the 4 task types in the FR text datasets. 36
4.5. Distribution of the target labels for the 4 task types i.e. Category, Color, Brand, Gender, in the UK Dataset. 37
4.6. Distribution of the target labels for the 4 task types i.e. Category, Color, Brand, Gender, in the DE Dataset. 38
4.7. Distribution of the target labels for the 4 task types i.e. Category, Color, Brand, Gender, in the FR Dataset. 39
4.8. Distribution of the target labels for the 4 task types i.e. Category, Color, Brand, Gender, in the UK Image Dataset. 41
4.9. Distribution of the target labels for the 4 task types i.e. Category, Color, Brand, Gender, in the DE Image Dataset. 42
4.10. Distribution of the target labels for the 4 task types i.e. Category, Color, Brand, Gender, in the IT Image Dataset. 43

5.1. Model Architecture of the Transformer. 46
5.2. Left: Inception-ResNet-v2 network architecture. Right: Schema of the Stem block in Inception-ResNet-v2. 48
5.3. Model Architecture of the BERT. 50
5.4. Training strategies employed for image based transfer learning using the Inception.Resnet-V2 model. 51

6.1. Accuracy scores for STL for the DE and UK tasks for both image and text based datasets. 59
6.2. F1 scores for STL for the DE and UK tasks for both image and text based datasets. 59
6.3. Accuracy scores for the two strategies used to implement TL on the image datasets. 61
6.4. F1 scores for the two strategies used to implement TL on the image datasets. . 61

6.5. Accuracy scores for TL for the DE and UK tasks for both image and text based datasets. 62
6.6. F1 scores for TL for the DE and UK tasks for both image and text based datasets. 62
6.7. Accuracy for all the text based datasets for Multi-task training. 63

7.1. Accuracy for all the approaches for the text based datasets. 67
7.2. F1-scores for STL and TL for the text based datasets. 67
7.3. Accuracy for all the approaches for the image based datasets. 68
7.4. F1-scores for all the approaches for the image based datasets. 68

List of Tables

4.1. Table containing samples from the text datasets. 30
4.2. Table showing the statistics of the text based datasets. 33
4.3. Table showing the statistics of the image based datasets. 40

6.1. Table depicting the details of the GPU used for the experiments. 54
6.2. Hyperparameters used for the text based experiments. 55
6.3. Hyperparameters used for the image based experiments. 55

7.1. Table containing the Accuracy and F1-scores for all the text based models. . . 64
7.2. Table containing the Accuracy and F1-scores for all the image based models. . 65

A.1. Detailed samples from the FR dataset 73
A.2. Detailed samples from the UK dataset. 74
A.3. Detailed samples from the DE dataset. 75

Bibliography

[1] Archive.org. *Wikipedia corpus download page*. 2017. URL: https://archive.org/download/enwiki-20171201 (visited on 06/07/2019).

[2] J.Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. *ImageNet: A Large-Scale Hierarchical Image Database*. IEEE Computer Vision and Pattern Recognition (CVPR), 2009.

[3] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu. *A Survey on Deep Transfer Learning*. arXiv:1808.01974v1, 2018.

[4] emarketer.com. *Worldwide Retail Ecommerce Sales Will Reach $1.915 Trillion This Year*. 2016. URL: https://www.emarketer.com/Article/Worldwide-Retail-Ecommerce-Sales-Will-Reach-1915-trillion-This-Year/1014369 (visited on 06/07/2019).

[5] M. Campbella, A. H. Jr, and F.-h. Hsu. *Deep Blue*. Elsevier Artificial Intelligence 134 (2002) 57–83, 2002.

[6] J. Manyika, M. Chui, B. Brown, R. D. Jacques Bughin, C. Roxburgh, and A. H. Byers. *Big data: The next frontier for innovation, competition,and productivity*. McKinsey Global Institute, 2011.

[7] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

[8] A. Maier, C. Syben, T. Lasser, and C. Riess. *A Gentle Introduction to Deep Learning in Medical Image Processing*. arXiv:1810.05401v2, 2018.

[9] A. Krizhevsky. "Learning Multiple Layers of Features from Tiny Images". In: *University of Toronto* (May 2012).

[10] Skymind.ai. *A Beginner's Guide to Multilayer Perceptrons (MLP)*. URL: https://skymind.ai/wiki/multilayer-perceptron.

[11] J. M. Nazzal, I. M. El-Emary, and S. A. Najim. *Multilayer Perceptron Neural Network (MLPs) For Analyzing the Properties of Jordan Oil Shale*. World Applied Sciences Journal 5 (5): 546-552, 2008.

[12] Y. LeCun, K. Kavukcuoglu, and C. Farabet. *Convolutional networks and applications in vision*. IEEE, 2010.

[13] D. Scherer, A. Müller, and S. Behnke. *Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition*. Springer, 2010.

[14] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. *How transferable are features in deep neural networks?* Curran Associates, Inc., 2014.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton. *ImageNet Classification with Deep Convolutional Neural Networks*. NIPS, 2012.

[16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. *Gradient-based Learning Applied to Document Recognition*. IEEE, 1998.

[17] E. Culurciello. *The fall of RNN / LSTM.* URL: `https://towardsdatascience.com/the-fall-of-rnn-lstm-2d1594c74ce0`.

[18] *Understanding LSTM Networks.* URL: `http://colah.github.io/posts/2015-08-Understanding-LSTMs/`.

[19] J. Brownlee. *A Gentle Introduction to Transfer Learning for Deep Learning, Machine Learning Mastery*. 2017.

[20] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. *Natural Language Processing (Almost) from Scratch*. Journal of Machine Learning Research, 2011.

[21] D. Sarkar. *A Comprehensive Hands-on Guide to Transfer Learning with Real-World Applications in Deep Learning- Towards Data Science*. 2018.

[22] Y. Zhang and Q. Yang. *A Survey on Multi-Task Learning*. arXiv:1707.08114v2, 2017.

[23] K. Hashimoto, C. Xiong, Y. Tsuruoka, and R. Socher. *A joint many-task model: Growing a neural network for multiple NLP tasks*. arXiv preprint arXiv:1611.01587, 2016.

[24] S. Ruder. *An Overview of Multi-Task Learning in Deep Neural Networks*. arXiv:1706.05098v1, 2017.

[25] K. Weiss, T. M. Khoshgoftaar, and D. Wang. *A survey of transfer learning*. 2016.

[26] A. Ramcharan, K. Baranowski, P. McCloskey, B. Ahmed, J. Legg, and D. P. Hughes. *Using Transfer Learning for Image-Based Cassava Disease Detection*. arXiv preprint arXiv:1707.03717v2, 2017.

[27] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi. *Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning*. arXiv:1602.07261v2, 2016.

[28] M. Lagunas and E. Garces. *Transfer Learning for Illustration Classification*. arXiv preprint arXiv:1806.02682v1, 2018.

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

[30] H. S. Yunhui Guo, A. Kumar, K. Grauman, T. Rosing, and R. Feris. *SpotTune:Transfer Learning through Adaptive Fine-tuning*. arXiv preprint arXiv:1811.08737v1, 2018.

[31] T. Semwal, G. Mathur, P. Yenigalla, and S. B. Nair. *A Practitioners' Guide to Transfer Learning for Text Classification using Convolutional Neural Networks.* arXiv preprint arXiv:1801.06480, 2018.

[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. *Attention Is All You Need*. arXiv:1706.03762v5, 2017.

[33] *Transformer: A Novel Neural Network Architecture for Language Understanding*. 2017. URL: `https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html`.

[34] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. *Deep contextualized word representations*. arXiv preprint arXiv:1802.05365v2, 2018.

[35] J. Alammar. *The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning)*. 2018. URL: http://jalammar.github.io/illustrated-bert/.

[36] J. Howard and S. Ruder. *Introducing state of the art text classification with universal language models*. 2018. URL: http://nlp.fast.ai/classification/2018/05/15/introducting-ulmfit.html.

[37] J. Howard and S. Ruder. *Universal Language Model Fine-tuning for Text Classification*. arXiv preprint arXiv:1801.06146v5, 2018.

[38] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. arXiv preprint arXiv:1810.04805v2, 2018.

[39] Ł. Kaiser, A. N. Gomez, N. Shazeer, A. Vaswani, N. Parmar, L. Jones, and J. Uszkoreit. *One Model To Learn Them All*. arXiv preprint arXiv:1706.05137v1, 2017.

[40] L. D. Consortium. *wsj1 complete*. Linguistic Data Consortium, Philadelphia, vol.LDC94S13A, 1994.

[41] T.-Y. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. *Microsoft COCO:common objects in context*. 2014.

[42] M. P. Marcus, B. Santorini, M. A. Marcinkiewicz, and A. Taylor. *Treebank-3 ldc99t42*. Philadelphia, Penn.: Linguistic Data Consortium, 1999.

[43] J. Wang, J. Tian, L. Qiu, S. Li, J. Lang, L. Si, and M. Lan. *A Multi-task Learning Approach for Improving Product Title Compression with User Search Log Data*. arXiv preprint arXiv:1801.01725v1, 2018.

[44] A. Benton, M. Mitchell, and D. Hovy. *Multi-task learning for mental health using social media text*. arXiv preprint arXiv:1712.03538, 2017.

[45] R. He and J. McAuley. *Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering*. 2016.

[46] J. McAuley, C. Targett, J. Shi, and A. van den Hengel. *Image-based recommendations on styles and substitutes*. SIGIR, 2015.

[47] J. McAuley. *Image-based recommendations on styles and substitutes*. 2015. URL: http://jmcauley.ucsd.edu/data/amazon/.

[48] H. Body. *How to Scrape Amazon.com: 19 Lessons I Learned While Crawling 1MM+ Product Listings*. 2016.

[49] H. Body. *public-amazon-crawler*. 2016. URL: https://github.com/hartleybrody/public-amazon-crawler.

[50] *Scraperapi*. URL: https://www.scraperapi.com/documentation.

[51] *Beautiful Soup Documentation*. URL: https://www.crummy.com/software/BeautifulSoup/bs4/doc/.

[52] A. Najmi. *amazon-crawler*. 2019. URL: https://github.com/aamnanajmi/amazon-crawler.

[53] J. L. G. Quintero. *How to strip html tags from a string in Python*. 2016. URL: https://medium.com/@jorlugaqui/how-to-strip-html-tags-from-a-string-in-python-7cb81a2bbf44.

[54] *re — Regular expression operations*. 2016. URL: https://docs.python.org/3/library/re.html.

[55] T. Young, D. Hazarika, S. Poria, and E. Cambria. *Recent Trends in Deep Learning Based Natural Language Processing*. arXiv:1708.02709v8, 2018.

[56] *sklearn.preprocessing.LabelBinarizer*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelBinarizer.html.

[57] *TORCHVISION.TRANSFORMS*. URL: https://pytorch.org/docs/stable/torchvision/transforms.html.

[58] G. B. Team. *tensor2tensor*. URL: https://github.com/tensorflow/tensor2tensor/blob/master/README.md.

[59] N. Kalchbrenner, L. Espeholt, K. Simonyan, A. van den Oord, A. Graves, and K. Kavukcuoglu. *Neural machine translation in linear time*. arXiv:1610.10099v2, 2017.

[60] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin. *Convolutional sequence to sequence learning*. arXiv:1705.03122v2, 2017.

[61] K. Loginova. *Attention in NLP*. URL: https://medium.com/@joealato/attention-in-nlp-734c6fa9d983.

[62] K. He, X. Zhang, S. Ren, and J. Sun. *Deep Residual Learning for Image Recognition*. arXiv:1512.03385v1, 2015.

[63] *Pytorch Tutorials*. URL: https://pytorch.org/tutorials.

[64] L. Mou, Z. Meng, R. Yan, G. Li, Y. Xu, L. Zhang, and Z. Jin. *How Transferable are Neural Networks in NLP Applications?* arXiv preprint arXiv:1603.06111, 2016.

[65] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. arXiv preprint arXiv:1810.04805v2, 2019.

[66] *pretrained-models.pytorch*. URL: https://github.com/Cadene/pretrained-models.pytorch.

[67] T. Dettmers. *Which GPU(s) to Get for Deep Learning: My Experience and Advice for Using GPUs in Deep Learning*. URL: https://timdettmers.com/2019/04/03/which-gpu-for-deep-learning/.

[68] *apex: A PyTorch Extension: Tools for easy mixed precision and distributed training in Pytorch*. URL: https://github.com/NVIDIA/apex.

[69] H. Mohammad and M. Sulaiman. *A Review on Evaluation Metrics for Data Classification Evaluations*. International Journal of Data Mining Knowledge Management Process, 2015.

[70] sebastianraschka. *What is the best validation metric for multi-class classification?* URL: https://sebastianraschka.com/faq/docs/multiclass-metric.html.