Rolv-Arild Braaten
August Bobakk Indal
Joakim Sæther

Improving classification of scanned documents using page stream segmentation and deep learning

Bachelor's thesis
May 2019

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Preface

This report is written in relation to our bachelor thesis at the Norwegian University of Science and Technology in Trondheim. All three team members have worked with machine learning as part of a course at NTNU earlier. During this course, we did a project concerning financial time series. After this project we wanted to continue working with machine learning for our bachelor thesis, which narrowed it down to a handful of projects. After a vote within the team, this project was chosen.

Firstly, we would like to thank our supervisor Ole Christian Eidheim for the guidance along the way, and for being genuinely interested in our research. Secondly, we want to thank Tore Johnsen and Per Inge Aas who have been our contacts at Geomatikk IKT. They have helped setting up a nice environment for us to work in, as well as providing data and domain knowledge. Finally, we would like to thank Freddy Wetjen from The National Library of Norway for providing some thoughts about possible approaches.

Assignment

The assignment was to use machine learning to improve upon Geomatikk IKT's existing classification solution for scanned documents. After meetings with our supervisor, we decided to focus on image classification, text classification and page stream segmentation, for which there are a multitude of different methods to explore.

The only real constraint made by Geomatikk IKT was that the solution had to run locally, as the data had some sensitive information that could not be distributed.

Because of the broad assignment, the team had to experiment with and compare different algorithms to find the most suitable.

Sammendrag

Denne rapporten sammenligner diverse metoder for å forbedre nøyaktigheten på klassifisering av norske kommunale dokumenter. Vi grupperer sammen sider ved hjelp av ulike modeller som ser etter relasjoner mellom sider, før vi kombinerer denne grupperingen med en eksisterende modell for klassifisering. Vi bruker også de eksisterende metodene ULMFiT og ResNet for tekst- og bildeklassifisering ved hjelp av dyp læring.

Resultatene viser at tekstklassifiseringen fra ULMFiT gir de beste resultatene på alle testede datasett, med en gjennomsnittlig forbedring på 30% i forhold til den nåværende løsningen. Den beste nøyaktigheten oppnådd var 91.1%. Vi ser likevel potensiale for fremtidig arbeid innenfor både ResNet og sidegruppering.

Abstract

This report compares a selection of methods for improving classification accuracy on Norwegian administrative documents. We group pages together by using multiple models that looks for continuities between pages, before combining this grouping with an existing classification model. We also use the existing deep learning methods ULMFiT and ResNet for text and image classification respectively.

We find that our ULMFiT text classification provides the best accuracy on all tested datasets, giving an average improvement of 30% to the current solution. The best accuracy achieved was 91.1%. However we see potential for future work involving both ResNet and page stream segmentation.

Contents

List of Figures

List of Tables

1 Introduction and relevance

This report is written in relation to a project at Geomatikk IKT in Trondheim. Geomatikk IKT scans large volumes of documents for their customers every single day in order to digitize and archive the material. To organize the archives, every page scanned needs to be assigned a category. This job is performed manually, which is a slow process taking up many hours of work.

The ultimate goal with this project is to automate this classification. Geomatikk IKT already utilizes a machine learning solution to aid in classification, but the model is not entirely accurate, so everything still needs to be double checked manually.

We will use the term "document" to signify an ordered collection of scanned images. A single image in such a document is called a "page".

Going by advice from the people doing this work manually, one logical next step is to improve this model by assigning the same category to pages that seem to belong together, but can not be classified on their own. To achieve this we were asked to create a model that checks whether two pages are connected. We will refer to this process as page stream segmentation (PSS).

Since their current model primarily uses automatically extracted text from the images of the scanned documents, another logical step is to start looking at other parts of the images, like specific shapes, textures or layouts. This could be used either as a supplement to PSS, or as a separate classifier.

Lastly, since their current method performs suboptimally, it is possible that their text classification solution does not hold up when compared to state-of-the-art methods. We would like to try to build a more updated text classifier using natural language processing.

To create a useful model to Geomatikk IKT, we would like to research multiple models, and compare their results to see the differences in performance. This means the research objectives of this report are:

- How does page stream segmentation affect classification performance?
- How does image classification perform on scanned administrative documents?
- How does text classification using natural language processing perform on scanned administrative documents?

Because the assignment is so vaguely defined, and consists mainly of experimenting, there are no documents specifying the requirements for the solution, nor are there any vision documents or system architecture documents.

Chapter 2 and 3 will look at the theory and previous work that is central to this report. Chapter 4 will describe the methods and technology used in the project in a manner that allows the reader to reproduce the results presented in Chapter 5. Chapter 6 and 7 will contain a discussion of our results as well as a conclusion and our recommendations for future work on the topic.

Acronyms and abbreviations

CNN Convolutional Neural Network

FN False Negative

FP False positive

MCC Matthews Correlation Coefficient

NN Neural Network

OCR Optical Character Recognition

PPV Positive Predicted Value

PSS Page Stream Segmentation

TN True Negative

TP True Positive

2 Theory

The following chapter contains all relevant theory to the work presented in this report.

The chapter will start off with some basic concepts in machine learning, before moving towards the theory that is more specific to the task at hand.

2.1 Supervised learning

Supervised learning is a collective term used to describe machine learning algorithms where both input and desired output data are provided. The fact that the desired output is provided during training of the model creates a basis to support future judgments on unseen data points [33]. The methods in which these algorithms learn are meant to mimic how humans learn.

There are two main weaknesses to this approach. Firstly, the model becomes bad at predicting data points that are unlike anything it has seen before. The class assignment of such points will be pretty much random. Secondly, the model needs to be trained on a large amount of data, which can be time consuming if the data is not easily available.

2.2 Unsupervised learning

Unlike supervised learning where the data is labeled with the appropriate output, unsupervised learning models have to learn the relationships between data points without guidance. This is done by letting the model's algorithms act on the data without prior training.

Since unsupervised models don't need to train on the data, these models can perform more complex processing tasks than supervised learning algorithms and need less data to be effective. On the other hand, unsupervised models can be more unpredictable than the alternate model, so the algorithm to use must be chosen carefully to fit the problem at hand. [34]

2.3 Datasets

When training a machine learning model it is common to split the original dataset into multiple smaller datasets.

The **training dataset** is the main proportion of the data. This data is used to train the model to recognize relevant patterns, and fit the model to make

judgments on future data.
The **validation dataset** is used during training to ensure that the model always performs well on data it has not been trained on. An advantage of using a validation set is early stopping. This means that the training can be stopped when the error on the validation set grows, and the previous model is chosen (the one with minimum error).
Finally the **test dataset** is used to evaluate the model, and give a measure of how well the model did. It is important that the model has not seen the data in the testing set, as this will give the best measure of how well the model will do on future unseen data.
There is no best way approach for dividing the data into training, validation and test datasets. It all depends on the problem at hand, and how much data is available.

2.4 Imbalanced datasets

In many machine learning applications the datasets will contain an imbalanced number of samples from each class. Such datasets are called **imbalanced datasets**. The problem with imbalanced datasets is that many classification algorithms are biased towards the majority class, which is not optimal considering the minority class often represents the most important concept to be learned. [28]

2.5 Neural networks

Neural networks (NN) are a class of supervised learning algorithms that are somewhat based on biological neural networks (brains). Like brains they contain "neurons" which can transmit signals between each other. Adjusting the strength of the connections between neurons enables neural networks to learn complex tasks, such as speech and image recognition, finance, and medical diagnosis. [6]

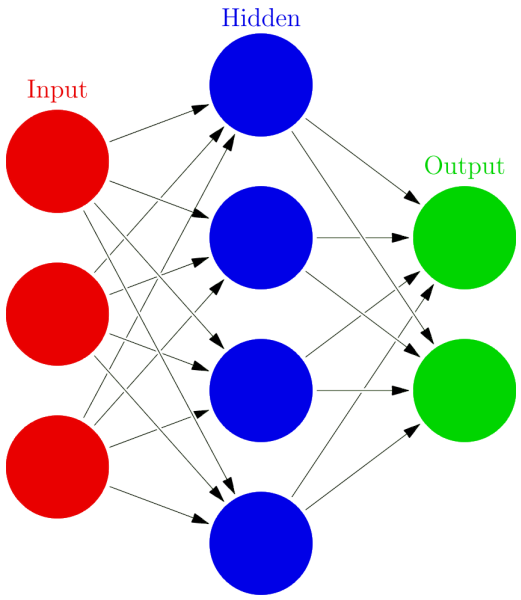

Figure 1: A simple neural network [17]

There are a few operations that can be done on neural networks in order to speed up the training process or to improve the accuracy.

2.5.1 Convolutional neural networks

Convolutional neural networks (CNN) are a special type of neural networks in which a mathematical operation called convolution is used to help the neural network learn better. Its benefit compared to a normal neural network comes from the fact that instead of just considering the values of each input individually, it also applies filters to consider all the inputs "around" it, and can then do that same filtering across the entire input space. It is commonly applied to analyzing visual imagery (photos and videos). [36]

2.5.2 Activation functions

All neural networks uses an activation function to activate its neurons. The activation function is what decides which criteria needs to be filled before a neuron fires. An important property of an activation function is that combinations of the function needs to be non-linear. If this property is not present, the neural network would behave just like a single-layer network,

because summing the layers would just give another linear function [44]. The most common activation functions will be presented below.

- **The sigmoid function** is given by:

$$A = \frac{1}{1 + e^{-x}} \tag{1}$$

  The main advantages of the Sigmoid-function is that it gives an analogue activation (not binary like a step-function) and the output of the function will always be between 0 and 1, compared to -∞ to ∞ for a linear function.

  The biggest issue with using the Sigmoid-function is that the gradient of the function "vanishes" for low and high values. This means that a big change in input produces a very small change in output. This is called *The problem of vanishing gradients* [45]. The result of this could be that the network refuses to learn further, or is drastically slow.

- **The softmax function** is given by:

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \text{ for } i = 1, \ldots, K \text{ and } \mathbf{z} = (z_1, \ldots, z_K) \in \mathbb{R}^K \tag{2}$$

  Which in words means that the function normalizes a set of real numbers such that each number is between 0 and 1 and their total sum is 1, meaning that it can be interpreted as probabilities. It is commonly used as the output of a classification model, where each index in the set corresponds to a specific class, and the number represents the probability that the input belongs to this class.

- **The rectified linear unit** (ReLU) is given by:

$$f(x) = max(0, x) \tag{3}$$

  The advantage of the ReLU is that the gradient never vanishes on the positive side (like in the Sigmoid function). The ReLU is also less computationally expensive than Softmax and Sigmoid, because it involves simpler mathematical operations.

The ReLU does however introduce a new problem. The gradient is always zero for negative values. This means neurons going into this state will stop responding to variations in input. This is called *The dying ReLU problem* [27].

There are variations to ReLU to avoid this issue by simply making the horizontal line non-horizontal by for example setting y = 0.01x for negative values of x. This adjustment is called Leaky ReLU. [27]

2.5.3 Dropout

Dropout is an operation that serves to reduce overfitting in neural networks. Individual nodes are "dropped out" of the net with probability (1 - p) or kept with probability p. Training is then performed on the reduced net before the dropped nodes are reinserted with their original weights. [40]

2.5.4 Batch normalization

When training a neural network you would usually normalize the input parameters in order to increase learning speed. This normalization-process can also be done for the layers of the neural network, which is called batch normalization.

By performing batch normalization, we allow for a higher learning rate, because we can be sure that there is no activation that have gone really high or low. Batch normalization also reduces overfitting because it adds some noise to the activation (similarly to dropout). [15]

2.5.5 Residual blocks

In a traditional neural network, each layer feeds into the next layer. In a network containing residual blocks, each layer feeds into the layers 2-3 hops away as well. The effect of this is essentially that the neural net has the option to skip layers that result in a reduction in accuracy. Figure 2 shows the flow of information through a residual block.

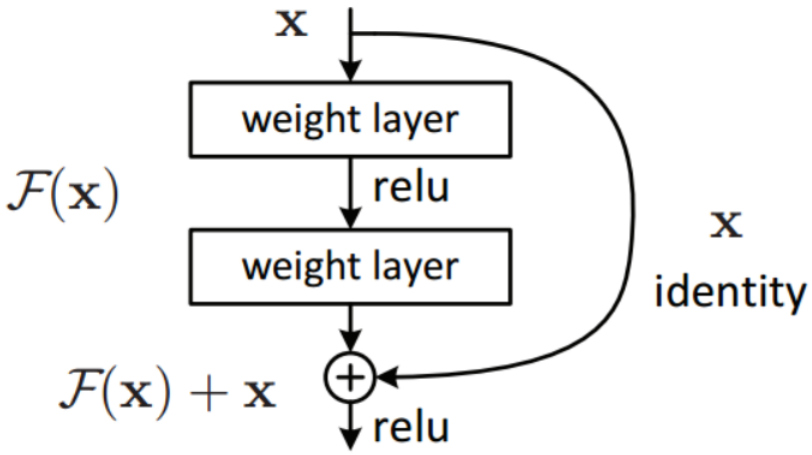

Figure 2: A single residual block. The network has the option to set the weights of the two layers to zero, and still have the input x on the other side of the block. [37]

Sahoo [37] presents a more in-depth explanation of residual blocks.

2.6 Training

Training is one of the most important aspects of supervised machine learning. A good training strategy can yield benefits both in pure accuracy, and through time savings.

2.6.1 Learning rate

Learning rate is a machine learning parameter that determines how much a neural network's weights should change depending on the gradient of the loss function. The learning rate is important, as a higher learning rate may lead to constantly overshooting local minima, while a lower learning rate will converge very slowly, meaning more time has to be spent training [46]. Figure 3 visualizes and compares various learning rates.

2.6.2 Epochs

An epoch is a single full cycle through the training dataset during training. Epochs are in some sense a way to compensate for low learning rate, since

taking multiple small steps will improve accuracy. However, more epochs mean more training time, and too many epochs could lead to overfitting, which is when the model adjusts itself excessively to fit the training data, and starts performing worse on unseen data. [39]

2.6.3 Optimizers

Optimizers are functions which tell the model how to change the weights depending on the result of the loss function. Different optimizers can produce wildly different results, and some are better for certain applications. Stochastic gradient descent, RMSProp, Adam, and AdaGrad are some of the most common optimizers used in machine learning today. [25]

2.6.4 Transfer learning

Transfer learning is a machine learning technique that involves training a model on one dataset, before performing specialized training on a different but related dataset. This allows for reduced training time, as the general training only needs to be performed once. [8]

2.6.5 Freezing

A common technique in deep learning is "freezing" layers, which prevents a layer's weights from being modified during training. This provides a speed-up due to reduced computation, and is commonly used in transfer learning for fine-tuning. [7]

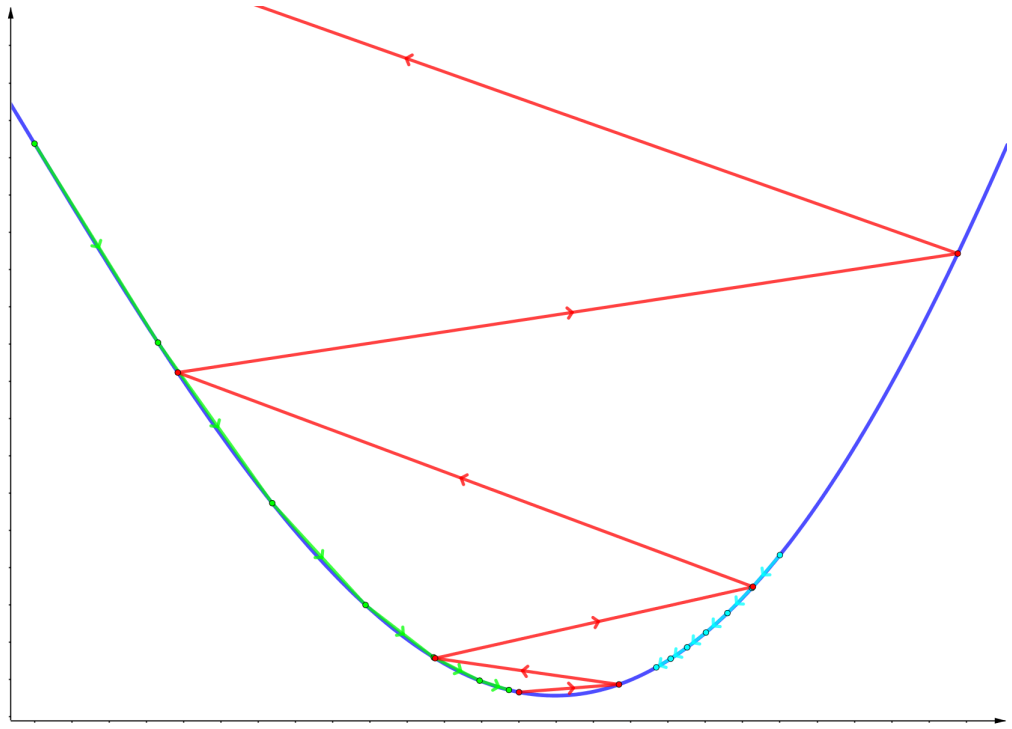

Figure 3: A highly simplified representation of how the result of the loss function (in dark blue) is affected by training. The three colors represent different learning rates, with the same number of epochs. The cyan points represent a low learning rate, and show that although accurate, the convergence is quite slow. The red points represent a high learning rate where the loss, despite starting closer to the minimum, diverges away. The green points represent the better learning rate of the three, where there is a compromise between speed and accuracy. It starts higher than both the others, and still ends closer to the minimum.

2.7 The curse of dimensionality

The curse of dimensionality is a phenomenon that arises when analyzing data in higher dimensions (often hundreds or thousands of dimensions). It comes from the fact that a linear increase in the number of features causes an exponential increase in the volume of the feature-space. The amount of data needed to support a statistically sound result also grows exponentially

with the dimensionality.

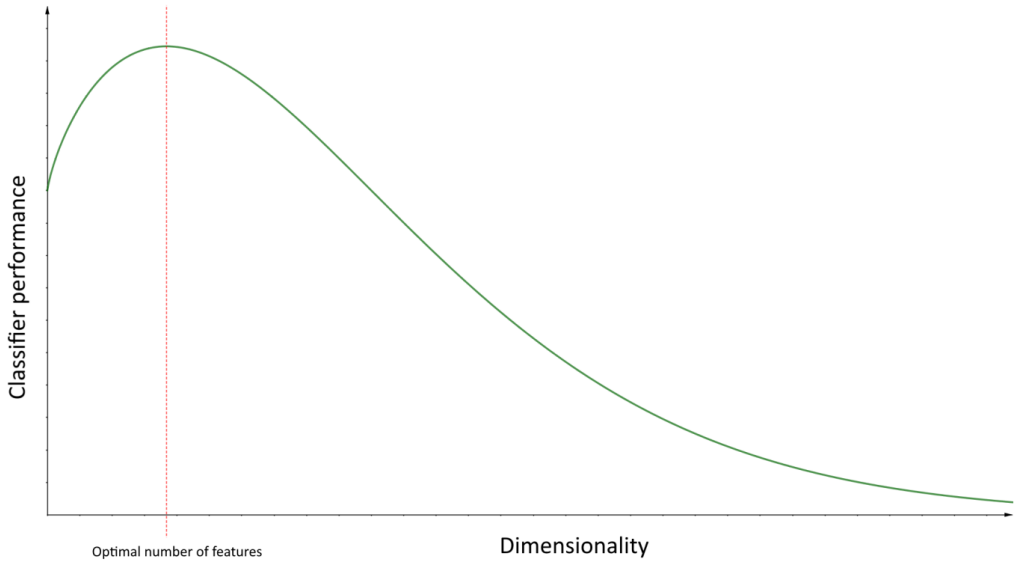

Figure 4: A simple representation of the performance of a machine learning model as dimensionality increases. With a fixed number of training samples, the performance of the model first increases, then decreases. This is knows as *Hughes phenomenon*. [3]

2.8 Dimension reduction

Complex machine learning problems can contain hundreds or even thousands of dimensions. As discussed in section 2.7 this could introduce some challenges. The solution to this is dimension reduction.
There are two main approaches to dimension reduction [1]:

- **Feature selection** tries to select the best combination of features from the original feature space. This involves removing features that have correlations with other features.
- **Feature extraction** attempts to combine the original features into new features to obtain a more relevant and compact feature space.

These approaches can be combined to reduce dimensionality even further.

2.9 Evaluating a model

When you have multiple algorithms solving the same problem, you will need a way to measure the algorithms against each other. As there are numerous evaluation algorithms depending on the task to be solved, this section will only cover the most important measures for this report.

2.9.1 Confusion matrix derivations

A confusion matrix (or error matrix) is a table layout that visualizes the performance of an algorithm. Each row in Figure 5 represents the instances in a predicted class while each column represents the instances in an actual class.

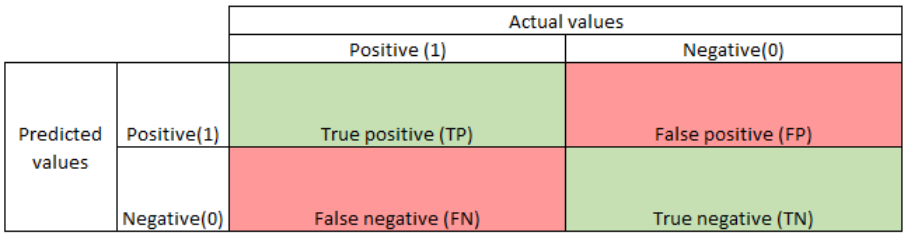

Figure 5: Confusion matrix

From a confusion matrix we can derive quite a few measures, but only a few are worth mentioning here:

- **Accuracy** is simply a measure of how often the model is correct. Accuracy is calculated as correct guesses divided by total number of guesses.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{4}$$

- **Positive Predicted Value** (PPV) gives an indication of how many of the positive predictions that are TP.

$$PPV = \frac{TP}{TP + FP} \tag{5}$$

- **Matthews Correlation Coefficient** (MCC) measures the correlation between the observed and predicted classification. MCC gives an

output describing how much better or worse the model is compared to random guessing. A correlation of 0 represents random guessing, while 1 and -1 represents perfect correlation and inverse perfect correlation respectively [29]. MCC is calculated as:

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad (6)$$

The greatest benefit of the MCC is how well it evaluates predictions on imbalanced datasets. Let us consider a simple case where 95 out of 100 cases belong to one class, while the last 5 cases belong to a second class. Let's say the model has the following confusion matrix:

	Class 1	Class 2
Class 1	90	4
Class 2	5	1

The model will score an accuracy of 91%, but the model still only predicts a single instance of class 2 correctly. The MCC will however punish the model with a score of only 0.14, which indicates a classifier that is close to random guessing. [10]

For further reading on confusion matrix derivations see [32].

2.9.2 Loss function

A loss function is used to evaluate how closely a model's predictions matches the correct answers. In a neural network the loss function is used for the gradient descent operations during training. There are a multitude of different loss functions, using a variety of different mathematical techniques, for example mean squared error, or sigmoid/softmax cross entropy. Contrary to accuracy, the loss functions are usually only used on the training data, and take the model's confidence into account. [31]

2.10 Text pre-processing

Before anything meaningful can be done with text data, the text might need to be processed in order to extract the relevant features, and remove noise from the text.[43] The following section will introduce some common methods for pre-processing text.

2.10.1 Tokenization

Tokenization is the process of breaking a string into tokens such as words, keywords and phrases. Breaking the text into tokens can be useful in order to compare different texts in terms of tokens.[43]

2.10.2 Stop word-removal

Stop word-removal is the process of removing words that does not relate to any particular topic, and add no additional information to a text string. These are words such as "the", "but" and "what".

Stop words are different depending on the situation they are used, and on the language being used. There are lists of general stop words for most languages online.[43]

2.10.3 Lowercase conversion

Lowercase conversion is the action of converting words into lowercase. For most applications it is preferred that words are treated as equal regardless of whether they are upper- or lowercase.[43]

2.10.4 Stemming

Stemming is the act of reducing derived words to their word stem or their root form. The idea is that the words "type", "types", "typed" and "typing" adds the same information to a text, and therefore can be treated as equal. Just like stop words, stemming depends on the language being used.[43]

2.11 Text classification

The goal of text classification is to assign different texts into different categories depending on their content. Unstructured data is everywhere around us, so the application of text classification is broad. Examples of applications can be organizing news articles by topics, or spam e-mail filtering.

This section will present some of the most common algorithms for performing text classification tasks.

2.11.1 Doc2Vec

To explain the Doc2Vec algorithm we first need to visit the Word2Vec algorithm. Word2Vec tries to capture not only the words of a text, but also the relation between words that have some kind of a connection [2]. For example it will give "king" and "queen" the same relation as "man" and "woman".

The Doc2Vec algorithm [9] is based on the Word2Vec algorithm. Where Word2Vec tries to represent the relation between words, the Doc2Vec algorithm instead captures the relation between documents numerically.

2.11.2 n-gram

The n-gram algorithm counts occurences of specific sequences of words/letters, with n denoting the length of the sequence [24]. For example, the word "banana" gives character unigrams: {a: 3, b: 1, n: 2}, bigrams: {an: 2, ba: 1, na: 2}, and trigrams: {ana: 2, ban: 1, nan: 1}.

2.11.3 FastText

FastText is an open source library created by Facebook to aid with word representations and text classification. The fastText library is in its simplest form, an extension to the Word2Vec algorithm (briefly described in Chapter 2.11.1). The key difference is that fastText uses n-grams, while Word2vec treats each word as an atomic entity.[23]

2.11.4 Universal Language Model Fine-tuning (ULMFiT)

ULMFiT is a transfer learning method that can be applied to any task in Natural language processing. A language model is trained to predict the next word of a sentence. This model can be trained on a big general text dataset, like Wikipedia, to "learn" the language. The language model can then be adjusted on a specific dataset with less training. After adjusting the language model for the specific dataset you can use it to create a classification model. The main advantages of ULMFiT is the reduced training time compared to alternate natural language processing methods that needs to be trained from scratch. [19]

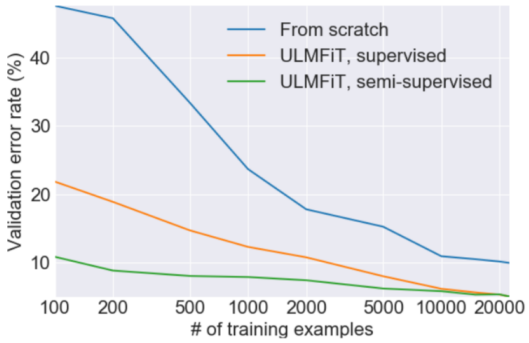

Figure 6: ULMFiT used on IMDB classification problem. From scratch is without transfer learning. Ulmfit supervised is pretrained on Wikipedia and adjusted on the training data. Ulmfit semi-supervised is the same as supervised except it also uses unlabeled data to adjust the language model. [19]

2.12 Image processing

When dealing with images in machine learning, one will often have to deal with a large number of dimensions. Having three dimensions (red, green and blue) for each pixel of the image will not work for high resolution pictures. Because of this we need some algorithms to extract the most relevant information out of a picture in order to scale the picture into a lower dimensionality. This section will provide some common algorithms for image processing.

2.12.1 Re-sizing

The most intuitive way to reduce the dimensionality of an image is by simply re-sizing it, either in number of pixels or in number of color channels. By reducing a picture to greyscale one can reduce the dimensionality by $\frac{2}{3}$, as each pixel will have one dimension instead of three.

2.13 Distance metrics

Any clustering method needs a way to evaluate the distance between two points in the feature space. To do this there exists different distance metrics

that all have different strengths and weaknesses. Strehl *et al.* [41] provide an in-depth analysis of a few metrics on a general basis, finding that the most popular metric, Euclidean Distance, performs considerably worse than cosine correlation and extended Jaccard measures in high dimensional space.

Anna Huang [20] specializes this towards text document clustering with similar findings.

3 Previous work

This section will provide an overview of the work that has already been done regarding page stream segmentation (PSS) and document classification.

3.1 Existing solution

Geomatikk IKT's existing solution uses Microsoft Azure to filter certain classes by image and obtain OCR information. This OCR is then used to train a fastText model for the rest of the classes.

3.2 Segmentation

Quite a few papers have been written about PSS in different forms, and most of them are shaped to fit a particular problem with a particular set of documents. These papers can generally be split into rule-based approaches and machine learning approaches.

3.2.1 Rule-based approach

The rule-based approach implements specific rules of the form "If X then do Y else if P then do Q". This approach performs well if the data is structural and contains only a few rules, but are sensitive to new unseen data. If the data is unstructured it would require a manual job to add new rules as the old rules fail.

Karpinski and Belaïd [26], and Daher *et al.* [12] present a rule-based approach to PSS. Daher *et al.* utilizes a set of rules to extract textural descriptors from two successive pages, then use these descriptors to distinguish between ruptures and continuity in the page stream. They do not mention the nature of the dataset used in the study.

Karpinski and Belaïd build further on the work of Daher *et al.* by adding descriptors for the structure of a page. The dataset used in this study consists of single-page and multi-page e-mails. The results section of the study is divided into single-page documents and multi-page documents, and shows the precision, recall and segmentation error for each set of documents. The best precision comes from the single-paged documents with 97%, while the multi-page documents post an accuracy of 81%.

3.2.2 Machine learning approach

The machine learning approach on the other hand deals well with previously unseen data, as it can make up these rules without the need for any human interference. Collins-Thompson and Niekolov [11] and Rusinol *et al.* [35] present two different machine learning approaches to the problem.

Collins-Thompson and Niekolov combine a layout descriptor with page numbering and information about headers and footers of the page to acquire a page separation accuracy of 95.7%. The algorithm utilized is a form of clustering where every page initially form its own cluster, before these clusters are increasingly grouped together.

Rusinol *et al.* use a bag-of-words approach to form a visual and a textual description of a single page. Different algorithms for fusing the pages into multipage documents are then applied, and evaluated. The results shows that the textural descriptor combined with a late fusion strategy achieves the best results with an accuracy of about 74%. It is however noted that the results might be influenced by the nature of the data, as the document classes are defined at a semantic level.

3.2.3 Comparing the two approaches

Finally Hamdi *et al.* [16] present a comparison of an existing rule-based approach and a machine learning approach using Doc2Vec. The data used contains different administrative documents with a big proportion of single-pages in the dataset. The dataset is divided into singlepage documents and multipage documents and the approaches are compared on the different datasets. The approaches are compared on segmentation errors, merging error, rate of complete documents properly structured and rate of good decisions predicted by the algorithms. The results show a slightly better accuracy for the rule-based method on the single-page dataset while the Doc2Vec method has a better accuracy on the multi-page documents.

3.3 Image classification

Document classification is a popular problem, due to an increasing number of documents, both in digital and paper form, that need to be categorized. Multiple papers have utilized the public dataset RVL-CDIP [18] for testing document classification models, as it provides a simple way to measure the

model against other proposed models. Some examples of such papers are:

- Das *et al.* [13] use two levels of transfer learning. The first layer consists of weights exported from a model pre-trained on the ImageNet dataset [14]. The second layer of transfer learning is trained to recognize segments of an image. The pedictions of the neural networks are then combined using using stacked generalization. This approach achieves an accuray of 92.21% on the RVL-CDIP dataset.
- Afzal *et al.* [5] use transfer learning from real images together with very deep neural network architectures such as GoogLeNet, VGG and ResNet. This achieves an accuracy of 90.97% on the RVL-CDIP dataset.
- Tensmeyer and Martinez [42] train CNNs with stochastic gradient descent, and experiment with different inputs for the models. They achieve an accuracy of 90.94% on the RVL-CDIP dataset.

3.4 Text classification

When it comes to text classification there are numerous datasets available for comparing new approaches to current state-of-the-art methods. The AG News corpus [21] is a collection of more than 1 million news articles. Some of the approaches for text classification on the AG News corpus is summarized below.

- State-of-the-art accuracy is achieved by Sachan, Zaheer and Salakhutdinov [38]. The method utilized is a bidirectional LSTM which differ from a regular LSTM by the fact that the signal can propagate backwards as well as forward in time. The accuracy achieved by this approach is 95.05%.
- A different approach by Howard and Ruder [19] using ULMFiT (discussed in Chapter 2.11.4) achieves an accuracy of 94.99%.
- Johnson and Zhang [22] propose an approach using a combination of supervised and semi-supervised learning. The approach utilizes a LSTM as well as convolutions trained on unlabeled data. This method achieves an accuracy of 93.43% on the AG News corpus.

- Joulin *et al.* [23] uses a BoW (Bag of Words) algorithm for text classification on the AG news dataset. This achieves an accuracy of 92.5% and is almost on par with the deep learning approaches presented above with far less computation time.

A more complete overview of different approaches for text classification can be found in the State-of-the-art table in [4].

4 Technology and method

The following sections will contain everything needed to reproduce the work done in this project. The development process of the team will be presented in addition to the relevant frameworks and the settings used in these frameworks.

4.1 Work process

The work in this project has been quite exploratory. Although the team has been working closely together through the entire process, it has been important for us to be able to work individually as well. To achieve this, all the team members have worked on different algorithms and different parts of the project.

The fact that the whole team has been sitting at Geomatikk IKT has almost removed the need for any meetings with our contacts at Geomatikk, as we have been communicating with them every day.

Even though none of the team members have been entirely responsible for any part of the project, we can distribute some main responsibilities:

- Rolv-Arild has had the main responsibility for the image model.
- August has had the main responsibility for the natural language modelling model.
- Joakim has had the main responsibility for the PSS models.

All team members took a course on applied machine learning the semester before beginning this project, so we knew a bit about the most common machine learning algorithms. Through the project we have learned more about the more specific machine learning tasks for text and image processing and PSS.

4.2 Development methodology

Because of the exploratory nature of the project, the team has not utilized a lot of specific development strategies. We did however set up a Discord server to write down ideas, keep track of which ideas had been tested, and to avoid stepping on each other's feet.

As we have been sitting closely together, communicating on a regular basis, the Discord server was all we needed to coordinate the work and keep track of the state of the project.

The project code was written using Python 3.6. We used the Jetbrains PyCharm IDE and Jupyter Notebook.

4.3 Datasets

Training and testing has been performed on the archives of Klepp kommune, Hamar kommune and Tromsø kommune during the project. From these three archives we have used four different datasets, all of which are summarized in Table 1 and Table 2.

Category	**Klepp Landbruk**	**Klepp**	**Hamar**	**Tromsø**	**SUM**
Annet	54 203	2 316	-	-	**56 519**
Ansvar og kontroll	-	1 101	729	-	**1 830**
Avtale	6 876	238	-	-	**7 114**
Avtale, Erklæring	2	-	150	-	**152**
Ferdigattest/brukstillatelse	-	143	63	-	**206**
Kart	-	3	-	-	**3**
Kart, tegning, foto	10 359	-	846	1 535	**12 740**
Klage	192	8	-	-	**200**
Korrespondanse	8	1 826	5 540	3 232	**10 606**
Målebrev	-	-	1 465	-	**1 465**
Null	136	-	42	-	**178**
Situasjonsplan, kart, skisse	6	582	-	-	**588**
Søknad	27 110	2 654	1 413	513	**31 690**
Tegning	-	-	678	-	**678**
Tegning, foto	2	-	-	-	**2**
Uttalelse	625	27	-	-	**652**
Uttale/Vedtak andre myndigheter	5	115	-	-	**120**
Vedtak	13 637	911	556	2 085	**17 189**
Vedtak (Hamar kommune)	-	-	1 001	-	**1 001**
SUM	**113 161**	**9 924**	**12 483**	**7 365**	**142 933**

Table 1: The frequency of the different categories in our datasets.

Length	Klepp Landbruk	Klepp	Hamar	Tromsø	SUM
1	18 143	1 753	1 754	1 264	**22 914**
2	19 962	1 408	1 141	856	**23 367**
3	4 591	412	382	448	**5 833**
4	3 229	306	284	279	**4 098**
5-9	3 085	298	325	281	**3 989**
10-19	529	50	92	20	**691**
20-49	82	10	28	0	**120**
50-99	6	1	5	0	**12**
101+	1	1	9	0	**11**
SUM	**49 628**	**4 239**	**4 020**	**3 148**	**61 035**

Table 2: Counts of different lengths of consecutive pages belonging to the same class.

The datasets are pretty similar in terms of the distribution of lengths of consecutive pages belonging to the same class. There are however some differences in the document classes in the different datasets. The dataset from Tromsø kommune contains only four different classes, and all the classes are well represented, containing more than 5% of the documents each. All of the other three datasets have more than 10 classes, where some of the classes are represented in less than 0.5% of the dataset.

In addition to these datasets, we also used the RVL-CDIP dataset [18] for training and testing of the image classification model. This dataset consists of 400 000 greyscale images of scanned documents, with 25 000 of each of 16 classes (see Figure 7). The dataset has 320 000 training images, 40 000 validation images and 40 000 test images. This dataset was useful as it gives the opportunity to use it anywhere, and train on more powerful hardware than was available at Geomatikk IKT.

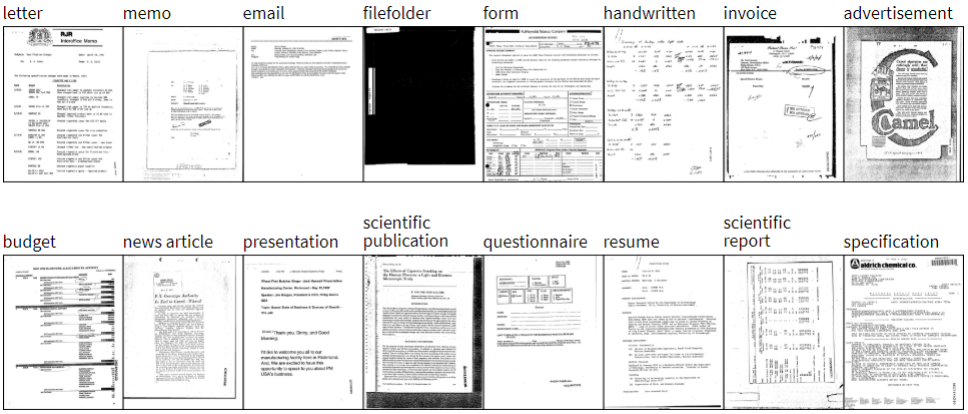

Figure 7: Examples of the different classes of the RVL-CDIP dataset.

4.3.1 OCR

The datasets contains a scanned image of the page, as well as OCR of the text on the page, supplied by Geomatikk IKT. In our experience the OCR generally reflects the text on the pages well, albeit with a few errors, particularly on handwritten text.

4.4 PSS models

For the PSS task we have created multiple models that look for relationships between two pages. The thought behind this is that while individually the models may perform decently, a combination of the models would perform even better than any single model. The different models we have utilized are described below.

- **Page-number**
  The page-number model looks for page numbers in the OCR-texts to check if the page-numbers in two pages correspond with each other. This is done by using various regular expressions to filter out the page numbers. The use of this model requires the language of the text to be known, in order to know which expressions to search for.

- **N-gram**
  The N-gram model looks for similar combinations of letters or words in

two texts. Two texts with similar words or similar sentences are more likely to be related. Because of the high dimensionality of the data handled by this model, we have chosen to use the Jaccard similarity (see 2.13). The N-gram model has been used both with and without pre-processing (lowercase conversion, stemming, stop-word removal and tokenization).

- **Gensim**
  Gensim is an open source library for natural language processing, which uses statistical machine learning [47]. It involves implementations of fastText, Word2Vec and Doc2Vec algorithms. The Gensim model performs a similar task to the n-gram model, and also uses the Jaccard similarity.

- **Image-size**
  The image-size model looks at the similarity in the width and height of two scanned pages, giving a higher similarity for pages of equal size.

- **Classifier prediction**
  The classifier prediction model looks for similarities in the prediction of Geomatikk IKT's existing classifier. The classifier gives a probability for each class for each page. Due to the low dimensionality of this data, these values are compared using Euclidean distance.

The threshold of all models are initially tuned by using a small proportion of the datasets as a sample. For the smaller datasets we have used 10 % of the dataset for training (about 800 - 1 100 pages), and the rest for testing. For the bigger dataset (Klepp Landbruk) we have used 1% for training (about 1 100 pages), and the rest for testing. Using about 1 000 pages is sufficient, as the threshold of the models converge quite quickly. This also leaves more data for testing purposes.

4.5 Image classification

Our image classification model was built using TensorFlow Keras' built-in ResNet50 implementation, initialized with weights from pretraining on the ImageNet dataset [14], with average pooling. We also replaced the 1000 class output layer with our own (softmax). The model was then trained on the

RVL-CDIP dataset with a batch size of 8, using an image size of 512x362, before specialized training on each of Geomatikk IKT's datasets were performed with batch size of 1 due to memory restrictions. The learning rate was initially 1e-5 in all cases, and was halved when the loss stopped decreasing, until reducing the learning rate had no effect at which point the training stopped. The training data was also shuffled before using TensorFlow's Dataset shuffle function with a buffer size of 1000.

4.6 Text classification

We tokenized each word from the text from all the documents. A few documents did not contain any text from the OCR and was removed from the dataset. Some labels were misspelled and were corrected.

The text classification model was built using the fast.ai library's language model and text classifier. We downloaded weights for the language model trained on the Norwegian Wikipedia, found on GitHub [30]. We then fine-tuned the language model on Geomatikk IKT's datasets individually. During training we started by freezing everything except the last layer, training for 1 epoch, then unfreezing everything for the remainder of the training. To find the initial learning rate we used the fast.ai learning rate finder. The learning rate finder plots loss over learning rate and we used the learning rate where the gradient is the steepest in negative direction.

After training the language model we used it to make a classification model. We froze all except the last layer during the first epoch, and unfroze one more layer each epoch, while also decreasing learning rate, until the accuracy stopped increasing.

5 Results

The following sections will contain the results from our research. We will first dive into the PSS task, before combining it with the existing classification model. We will then look into the performances of new ideas for both image classification and classification by natural language modelling.

5.1 Page stream segmentation

This section will provide the results of the various models from our research on PSS.

5.1.1 AJR

Before going into the PSS results it is important to have a concept of what a good PSS is. For the PSS to be of any practical use for Geomatikk IKT, it is important that the solution does not give too many false positives. However it is also important to give as many true positives as possible. In order to strike a balance between TP and FP, we have introduced a new metric we have called the AJR. The AJR is given by

$$AJR = MCC \cdot PPV^{\frac{\ln 0.5}{\ln 0.95}} \tag{7}$$

Plotting the AJR for various thresholds will produce a curve where the maximum AJR represents the threshold with the best balance between MCC and PPV. The most useful number for our application is the PPV. The PPV does however tend to favor a threshold that allows for very few TPs. Because of this we have balanced it out by using the MCC.

5.1.2 Individual models

The PSS was done by using multiple models (see 4.4), and then combining these models to form a "model of models". The results of each individual model, as well as the results of the combined model, are presented in Table 3.

	Klepp		Klepp L.		Hamar		Tromsø	
Model	TP	FP	TP	FP	TP	FP	TP	FP
N-gram	973	38	16 567	1 105	1 643	132	1 294	131
Gensim	525	2	10 936	882	1 096	153	1 289	154
Page Number	348	0	631	1	66	0	7	0
Image Size	1617	100	25 017	1 011	1844	90	669	70
Classifier[1]	NA	NA	4 223	176	3 047	276	904	49
Combined model	2478	134	36 804	2 484	3 410	279	1987	258
PPV	94.9%		93.7%		92.4%		88.5%	

[1] Full classifier information not available for all datasets.

Table 3: Results of the various PSS models. The combined model represents the combination of all the individual models, and the PPV is the positive predicted value of the combined model.

Because of the overlap between the individual models, the sum of all the individual models does not add up to the sum of the combined model in Table 3. The amount of overlap ranges from about 25% for the Klepp dataset to almost 100% for the Tromsø dataset. We can see that the combined model makes more guesses than each individual model with very little loss in accuracy.

5.1.3 Combining PSS and classification

For combining our PSS model with Geomatikk IKT's existing classification, we propose four different approaches. We present the results of these four approaches in Table 4.

- **Naive combination** (NC): For each page segment given by the PSS model we assign the most guessed class (given that the most guessed class is guessed more than 1 time) of the classifier to the entire segment. If the most guessed class is NULL, we assign the second most guessed class.

- **Naive combination with reservation** (NCR): For each page segment given by the PSS we assign the most guessed class of the classifier to the entire segment if the most guessed class is guessed in more than 50% of the segment. If the proportion is less than 50%, the classification remains untouched.

- **Naive combination with preprocessing** (NC+): Same approach as NC, but with preprocessed text for the n-gram model.
- **Naive combination with reservation and preprocessing** (NCR+): Same approach as NCR, but with preprocessed text for the n-gram model.

	Klepp	Klepp L.	Hamar	Tromsø	Average
Existing classifier	48.3%	53.2%	**56.9%**	**73.4%**	**58.0%**
NC	48.3%	53.1%	56.4%	72.2%	57.5%
NCR	**48.4%**	53.2%	56.5%	72.3%	57.6%
NC+	**48.4%**	53.2%	55.8%	72.2%	57.4%
NCR+	**48.4%**	**53.3%**	56.0%	72.3%	57.5%

Table 4: Results of the combination of the PSS model and Geomatikk IKT's existing classification with the different combination schemes mentioned above.

	Klepp	Klepp L.	Hamar	Tromsø	Average
NC	1.6%	4.0%	6.8%	7.8%	5.1%
NCR	1.4%	3.6%	5.8%	6.6%	4.4%
NC+	1.3%	3.1%	10.1%	7.5%	5.5%
NCR+	1.2%	2.8%	8.4%	6.5%	4.7%

Table 5: Amount of predictions changed by the PSS-model rounded to one decimal.

Table 4 and Table 5 present the results of the combination. Even though the PSS-model changes as much as 10% of the classifier's guesses, the accuracy does not change much more than 1% in any case.

5.2 Classification

In addition to the existing classification model we propose two new classification models. Pretraining on the RVL-CDIP dataset achieves an accuracy

of 90.6% on that dataset, which is close to the state of the art methods mentioned in Chapter 4.4. Our models were trained/validated using the first 80% of the datasets, and all models were tested on the last 20% of each dataset. The test results are presented in Table 6.

	Klepp	Klepp L.	Hamar	Tromsø	Average
Existing classifier	10.8%	62.4%	69.7%	71.1%	53.4%
ULMFiT	81.7%	91.1%	77.8%	88.2%	84.7%
ResNet	40.8%	63.8%	37.6%	68.9%	52.8%

Table 6: Results of the proposed classification models and Geomatikk IKT's existing model.

The accuracy of the existing classifier varies quite a bit from dataset to dataset. For Klepp Landbruk, Hamar and Tromsø, this can be explained by the nature of the datasets as well as the amount of training performed on the various datasets. The low accuracy on the Klepp dataset however, comes from the fact that the existing classifier predicts NULL for most of the instances in the test dataset. A NULL-prediction means that the classifier is not certain enough of its prediction to assign a class to the page.

To get a more accurate view of what the strengths and weaknesses of the models are, it is useful to look at the confusion matrix. Figure 10, fig. 8 and fig. 9 shows the confusion matrix for the various models on the dataset from Klepp landbruk. The numbers along the diagonals shows the number of correct guesses in each category.

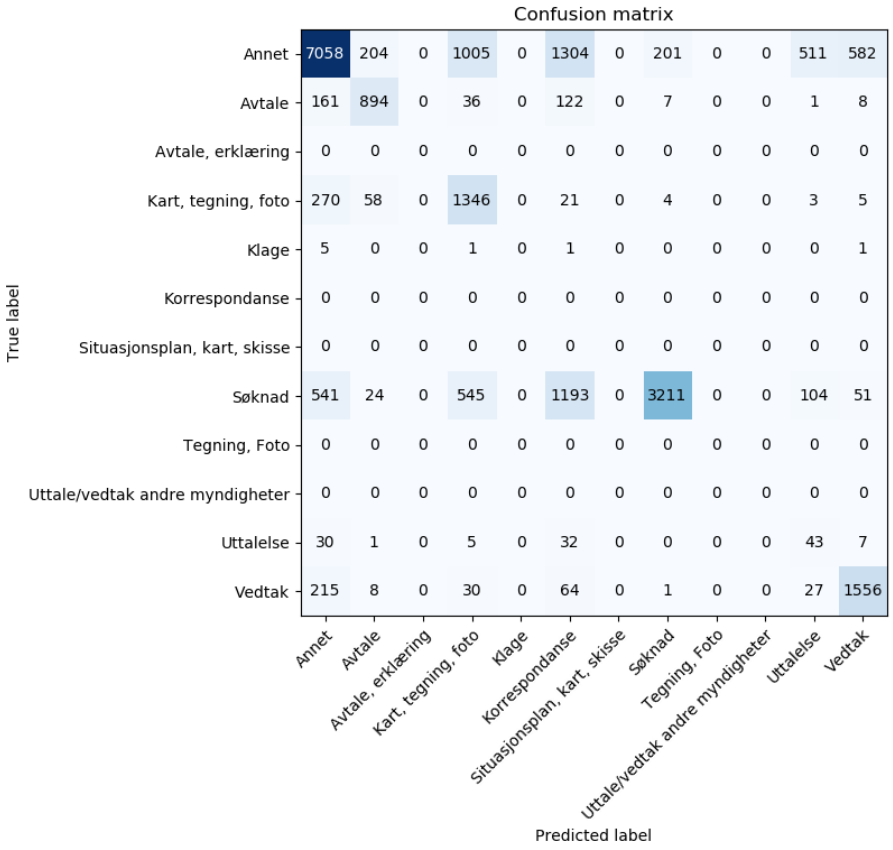

Figure 8: Confusion matrix for the existing classifier of Geomatikk.

We can see that the "Annet"-class, containing everything that doesn't fit elsewhere, is responsible for most of the errors of the ULMFiT model (Line 1 and column 1, fig. 8). We can also see a few dark colors along the line corresponding to "Søknad", which means that the classifier guesses other categories, when the page is actually a "Søknad". In addition to these two observations, it is worth to note that the testing dataset contained no examples of the class "Korrespondanse", while the existing model assigns this class to more than 2 500 pages.

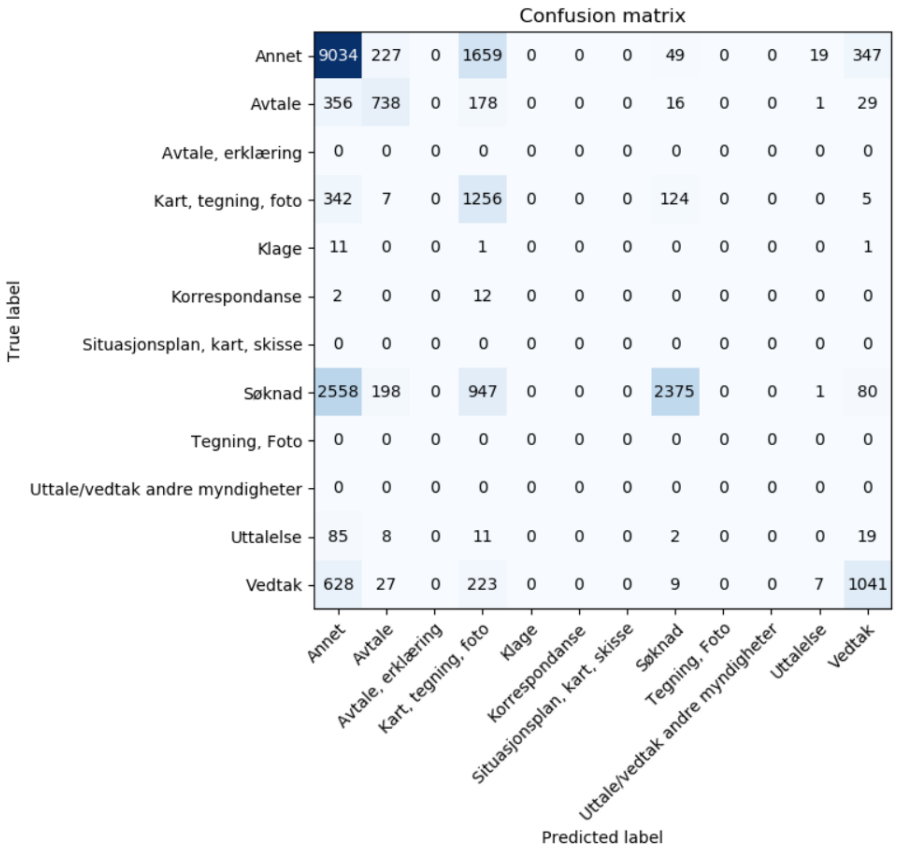

Figure 9: Confusion matrix for the ResNet model.

The ResNet model, just like the existing classifier, misses quite a few of the class "Søknad". Apart from that, most of the errors come from incorrectly guessing "Annet" or "Kart, tegning, foto". It has decent precision on "Søknad" and "Vedtak", but its recall isn't high on any of the classes compared to the ULMFiT model.

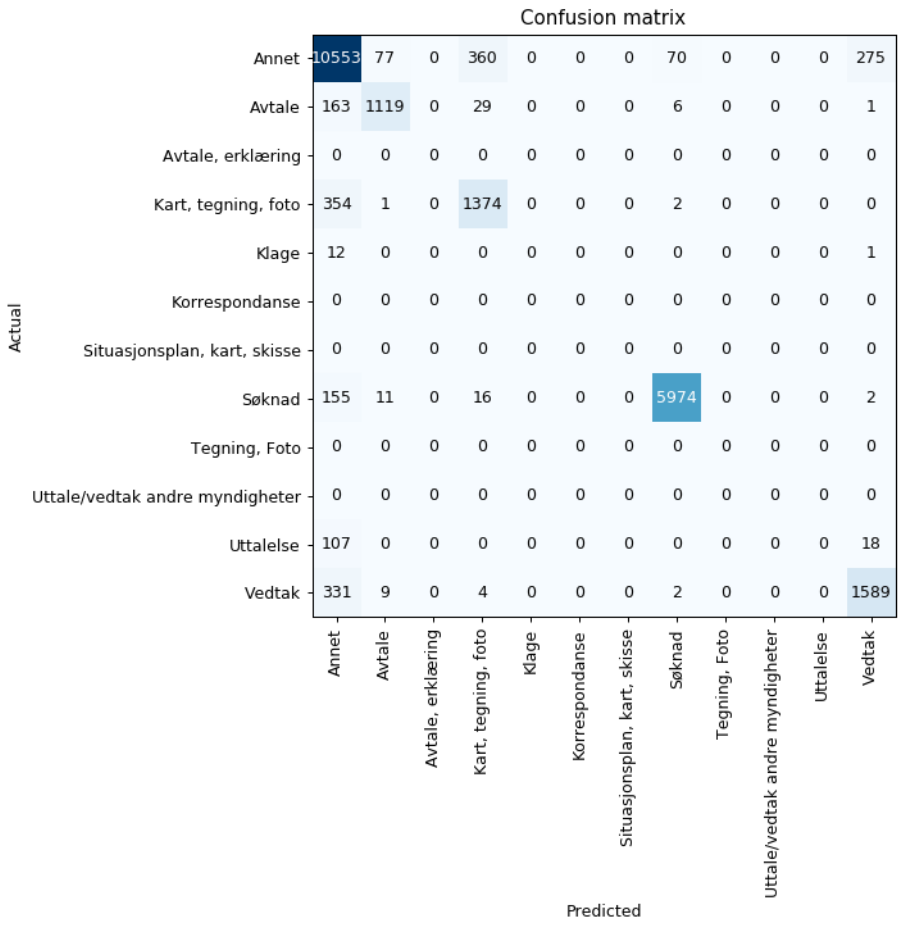

Figure 10: Confusion matrix for the ULMFiT model.

Just like the existing classifier, the ULMFiT model misguesses the most when it comes to the "Annet" class. Apart from this, the highest number of errors are found when the model guesses "Kart, tegning, foto", while the page is an "Avtale".

6 Discussion

The following chapter will provide a discussion of the results presented in the previous chapter, including sources of error. First, we will discuss the results from the page stream segmentation before moving on to the classification.

6.1 Page stream segmentation

The results from the PSS showed very little improvement to the classification accuracy, although the PPV of the PSS itself generally lies above 90% (Table 3). Intuitively one would think that the PSS would show better results combined with a good classification than with a bad classification. However, looking at Table 4, the PSS seems to do worse as the accuracy of the classification increases. Another interesting point is that the PSS model performs worse as the proportion of changed predictions increases (see Table 5).

The simplest explanation for these observations is that the PSS model correctly groups together the same pages that the classification has already predicted correctly. Because of the requirement for a high PPV in the solution, the proposed PSS model makes predictions for less than half of the data. It is likely that the pages that are easy to group correctly together are also the ones that are easy to classify. This means that both models performs well on the same pages.

Looking back at the previous work section, both our dataset and our proposed approach is most alike a combination of the two approaches presented by Hamdi *et al.*, but the results are still tough to compare. Although both studies are done on administrative documents, there are too many unknowns related to the similarities and differences of the datasets, to draw any conclusions.

6.1.1 Sources of error

During our research in page stream segmentation there have emerged a couple of problems that could have affected the results of out study. The list below summarizes these problems.

- The pages in the datasets used in the study have labels for which administrative case the pages are related to and for which class the page belongs to. This however, is not enough to determine whether two

pages belongs together. If a document has successive instances of a class, all of these pages will be treated as related even though the pages could be from different groups.

- Some of the datasets contains NULL-classes that could belong to any class. Such pages will only be treated as related if the next page is also marked with the class NULL. Because such cases will appear in production as well, we have chosen not to remove these pages from our datasets.

- The combination schemes proposed in this research are quite simple and could possibly be replaced by more advanced approaches to increase accuracy. This would require some research to be done.

Despite these issues, we believe that our results in page stream segmentation are representative for what would be observed in a real world setting.

6.2 Classification

The greatest proportion of failed guesses from all of the three classifiers are related the "Annet" class. Taking a closer look at the "Annet" class, one can discover that the class contains all the pages that does not naturally belong to any other classes. This gives a logical explanation to this observation, as the class contains a broad selection of different documents.

ULMFiT scored higher accuracy than the existing classifier from Geomatikk IKT on all datasets. On the largest dataset (Klepp Landbruk), we trained on around ten times as much data compared to the other three datasets. This is probably why we got the highest accuracy on that dataset. If that is the case, the accuracy could be further improved by providing more training data. On the other datasets, the results showed an increase in accuracy compared to Geomatikk's existing model even though we trained on less than 10 000 pages (where the existing model trained on 80 000).

Comparing our text classification to the results from the state-of-the-art table for the AG news dataset (mentioned in Chapter 3.4), we can see some similarities. In both cases, the ULMFiT algorithm has outperformed the FastText algorithm. In the AG news dataset, the difference was only 2.5%, but we have found a difference of about 30%. There might be multiple reasons for this gap, that are mentioned in the next section.

The ResNet model on the other hand, only scored a higher accuracy than the existing classifier on two of the datasets. It had an average accuracy on par with the existing classifier mostly because of the bad performance of the existing classifier on the Klepp dataset. The amount of training data used for the ResNet model does however seem to have a positive correlation to the accuracy, meaning that further increasing the amount of labelled samples could prove to increase the accuracy even further.

Both the ResNet model and the existing classifier misguesses on a big proportion of the "Søknad" class. The ResNet model assign the "Søknad" class to either "Annet" or "Kart, tegning foto", while the existing classifier assign them to either "Annet", "Kart, tegning, foto" or "Korrespondanse".

This difference seems to come from the difference in the nature of the training samples. The ResNet model did not have more than 8 samples of "Korrespondanse" in the training data, while the existing classifier of Geomatikk seems to have had quite a few. This is a good example of how much the training data could affect the result of a classification.

Comparing our results in image classification to the previous work done on the RVL-CDIP dataset gives quite a gap in accuracy. A possible explanation for this is mentioned in the next section.

Originally, we had intended to also combine the results from the image and text models into an ensemble model, but due to time constraints, as well as poor results on the image model compared to text, we decided not to try this. We think that with the improvements discussed above, the image model could be improved, and combining the two models might become viable.

6.2.1 Sources of error

Also in our classification research there has emerged some issues that could influence our results. These issues are listed below.

- Because our testing data contained sensitive information that could not be distributed outside Geomatikk IKT's systems, we could not utilize any more powerful computational power than what was provided by Geomatikk IKT for training and testing. This created some restrictions for the quite computationally expensive ResNet model, as we were not able to train it optimally. Better hardware would have allowed us to perform more hyperparameter tuning, which in turn could have led to improved accuracy of the ResNet model. We had to rely on best

guesses for the best ways to train the model, which is part of why we decided to pretrain using RVL-CDIP, so that we could use more powerful computers and spend less time training at Geomatikk IKT.

- The classes in our datasets are quite different to each other. Some classes appear very sparsely, while some are represented in more than half of the dataset (see Table 1). Many classes are not shared between datasets, making it hard to transfer what has been learned from one dataset to another. The classes can also be very similar, or even overlapping, and are not always well-defined. In a perfect world we would have standardized classes shared between all datasets, with equal distribution, where each class is somewhat visually distinct. This is more similar to the way the RVL-CDIP dataset is structured, in which case it is possible to get over 90% accuracy on images alone, and we theorize that adding text classification could increase that even further.

- Our comparison of the various models does not consider the amount of training data used by the models. Our classification models have, in some cases, used as few as 6 000 pages for training. The current model in use at Geomatikk is however trained on about 80 000 pages for each dataset.

- The splitting of our datasets was simply done by taking the first 80% and using it for training and the last 20% for testing. Scrambling the data before splitting it would make a test dataset that is more representative for the entire dataset. Looking at the variation in the accuracy on the Klepp dataset, we can see that this indeed had an impact. The existing classifier achieved 48.3% (Table 4) accuracy on the entire dataset, but only 10.9% (Table 6) on the test dataset.

- The Wikipedia language model used for transfer learning in ULMFiT is created from Norwegian Wikipedia. The vocabulary used in our model is taken from that model, which means we are missing nynorsk words as well as words misspelled from the OCR. The model also uses the top 60 000 words from Wikipedia and that may leave out some words that are more important in administrative documents, than in Wikipedia articles. Some documents in the datasets had no text and were discarded from testing.

7 Conclusion and future work

The following section will provide answers for the research questions presented in Chapter 1. We will also discuss some future work that could be done.

7.1 Page stream segmentation

How does page stream segmentation affect classification performance?

Our page stream segmentation algorithm performs well at identifying pages that belong together. Despite this, the combination with the existing classifier has very little, if slightly negative, impact. We think this may be due to overlap between the inputs of the classifier and the segmentation model.

7.2 Image classification

How does image classification perform on scanned administrative documents?

The image classification yields slightly better results than the existing classifier on some datasets, but due to being computationally expensive it was time-consuming to train and test with different parameters. Because our model was able to achieve near state-of-the-art accuracy on the RVL-CDIP dataset (90.6%), we think there may be gains in accuracy if some changes to the archiving system are made (Chapter 6.2.1), and more time was spent tuning the model.

7.3 Text classification

How does text classification using natural language processing perform on scanned administrative documents?

Our text classification model performed the best out of all tested models, achieving a maximum accuracy of 91.1% on the biggest dataset. The use of transfer learning makes the model less computationally expensive than both our image model, and the existing classifier of the firm, but we still believe that even higher accuracy could be achieved by further tuning the model.

7.4 Relevance

We have proposed two methods for document classification where one improved the current accuracy significantly, and also uses noticeably less data for training. We believe that the results of this research could contribute towards the ultimate goal of fully automating the task of classifying documents at Geomatikk IKT.

7.5 Future work

The list below contains our suggestions for future work to the solutions we have proposed.

- The schemes to combine PSS with document classification mentioned in this research is quite simple. Research regarding these schemes could prove beneficial to the accuracy of the combined model.
- Our proposed models for page stream segmentation achieve a high PPV by detecting continuities between successive pages (such as page numbering). A possible improvement could be a model that achieces a high PPV by detecting ruptures between two pages. We have attempted page sizes and background textures to perform this task without success, but other models might achieve greater results.
- To perform transfer learning you first need to train a general model on a big dataset. We used ImageNet, RVL-CDIP and Wikipedia as our general datasets. Geomatikk IKT has a huge amount of data from different archives. This can be used to create a general model for all the archives that can be further tuned on each archive. This might give higher accuracy and lower computation time using less data.
- Since we had limited time and computing power, we were unable to experiment much with our models. We would like to try things like different learning rates, optimizers, batch sizes or image sizes, adding more layers, other models such as DenseNet, freezing layers, and testing with and without pretraining.
- If the image model is improved we feel it may be worthwhile to combine the text and image models into an ensemble model. This should generate better guesses than the models can provide individually, since

they look at different aspects of the pages. We also see potential for integrating PSS into this model, as it may catch information about the page stream that the page itself may not contain.

- Due to the way Geomatikk IKT has structured their task, they have several people whose job it is to label data. We think this workload could be heavily reduced by using a principle called active machine learning. This means that the model is first trained to a certain accuracy on fully labeled data, then we start accepting its most confident guesses. The model can query the people who label the data when it is less confident, and train it into the model "actively". This would be especially viable if the model can be generalized to work on several archives.

References

[1] Introduction to dimensionality reduction. https://www.geeksforgeeks.org/dimensionality-reduction/. (Accessed on 2019/02/20).

[2] Vector representations of words. https://www.tensorflow.org/tutorials/representation/word2vec. (Accessed on 2019/05/07).

[3] Hughes phenomenon. http://37steps.com/2322/hughes-phenomenon/, 2013. (Accessed on 2019/02/20).

[4] Text classification on ag news. https://paperswithcode.com/sota/text-classification-on-ag-news, 2019. (Accessed on 2019/04/26).

[5] Muhammad Zeshan Afzal, Adreas Kölsch, Sheraz Ahmed, and Marcus Liwicki. Cutting the error by half: Investigation of very deep cnn and advanced training strategies for document image classification. Technical report, 2017.

[6] Deep AI. Neural network definition. https://deepai.org/machine-learning-glossary-and-terms/neural-network.

[7] Andrew Brock, Theodore Lim, J. M. Ritchie, and Nick Weston. Freeze-out: Accelerate training by progressively freezing layers. https://arxiv.org/abs/1706.04983, 2017.

[8] Jason Brownlee. A gentle introduction to transfer learning for deep learning. https://machinelearningmastery.com/transfer-learning-for-deep-learning/, 2019.

[9] Amar Budhiraja. A simple explanation of document embeddings generated using doc2vec. https://medium.com/@amarbudhiraja/understanding-document-embeddings-of-doc2vec-bfe7237a26da, 2018. (Accessed on 2019/05/07).

[10] Davide Chicco. Ten quick tips for machine learning in computational biology. *BioData Min*, pages 113–114. Tip 8.

[11] Kevyn Collins-Thompson and Radoslav Nickolov. A clustering-based algorithm for automatic document separation. 2002.

[12] Hani Daher, Mohamed-Rafik Bougelia, Abdel Belaïd, and Vincent Poulain D'Andecy. Multipage administrative document stream segmentation. 2014 22nd International Conference on Pattern Recognition.

[13] Arindam Das, Saikat Roy, Ujjwal Bhattacharya, and Swapan K. Parui. Document image classification with intra-domain transfer learning and stacked generalization of deep convolutional neural networks. Technical report, 2018.

[14] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[15] Firdaouss Doukkali. Batch normalization in neural networks. `https://towardsdatascience.com/batch-normalization-in-neural-networks-1ac91516821c`. (Accessed on 2019/04/10).

[16] Hamdi et al. Machine learning vs deterministic rule-based system for document stream segmentation. *IEEE*, pages 77–82, November 2017.

[17] Glosser.ca. Colored neural network. `https://nn.wikipedia.org/wiki/Fil:Colored_neural_network.svg`, 2013. (Accessed on 2019/02/21)(Digitally altered from original version).

[18] Adam W Harley, Alex Ufkes, and Konstantinos G Derpanis. Evaluation of deep convolutional nets for document image classification and retrieval. Technical report.

[19] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. Technical report, 2018.

[20] Anna Huang. Similarity measures for text document clustering. Technical report, The University of Wikateo, Hamilton, New Zealand, 2008.

[21] Mohammed H. Jabreel. Ag's news topic classification dataset. `https://github.com/mhjabreel/CharCnn_Keras/tree/master/data/ag_news_csv`, 2015. (Accessed on 2019/04/26).

[22] Rie Johnson and Tong Zhang. Supervised and semi-supervised text categorization using lstm for region embeddings. Technical report, 2016.

[23] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. Technical report, 2016.

[24] Shashank Kapadia. Introduction to language models: N-gram. `https://towardsdatascience.com/introduction-to-language-models-n-gram-e323081503d9`, 2019. (Accessed on 2019/05/07).

[25] Raimi Karim. 10 gradient descent optimisation algorithms + cheat sheet. `https://towardsdatascience.com/10-gradient-descent-optimisation-algorithms-86989510b5e9`, 2018. (Accessed on 2019/05/07).

[26] Romain Karpinski and Abdel Belaïd. Combination of structural and factural descriptors for document stream segmentation. 2016 12th IAPR Workshop on Document Analysis Systems.

[27] Danquing Liu. A practical guide to relu. `https://medium.com/tinymind/a-practical-guide-to-relu-b83ca804f1f7`, 2017. (Accessed on 2019/05/07).

[28] Victoria López, Alberto Fernández, Salvador García, Vasile Palade, and Francisco Herrera. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Elsevier*, pages 113–114, 2013.

[29] B. W. Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Elsevier*, 405:442–451, 1975.

[30] Jens Dahl Møllerhøj. Scandinavian ulmfit. `https://github.com/mollerhoj/Scandinavian-ULMFiT`, 2019. (Accessed on 2019/04/25).

[31] Ravindra Parmar. Common loss functions in machine learning. `https://towardsdatascience.com/common-loss-functions-in-machine-learning-46af0ffc4d23`, 2018. (Accessed on 2019/02/20).

[32] David M. W. Powers. Evaluation: From precision, recall and f-factor to roc, informedness, markedness & correlation. Technical report, Flanders University of South Australia, December 2007.

[33] Margaret Rouse. Supervised learning. `https://searchenterpriseai.techtarget.com/definition/supervised-learning`, 2016. (Accessed on 2019/02/20).

[34] Margaret Rouse. Unsupervised learning. `https://whatis.techtarget.com/definition/unsupervised-learning`, 2016. (Accessed on 2019/02/20).

[35] Marc¸al Rusinol, Dimosthenis Karatzas, Andrew D. Bagdanov, and Josep Llados. Multipage document retrieval by textual and visual representations. *IEEE*, 2012.

[36] Sumit Saha. A comprehensive guide to convolutional neural networks — the eli5 way. `https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53`, 2018. (Accessed on 2019/02/20).

[37] Sabyasachi Sahoo. Residual blocks — building blocks of resnet. `https://towardsdatascience.com/residual-blocks-building-blocks-of-resnet-fd90ca15d6ec`. (Accessed on 2019/04/08).

[38] Devendra Singh Sanchan, Manzil Zaheer, and Ruslan Salakhutdinov. Revisiting lstm networks for semi-supervised text classification via mixed objective function. Technical report, 2019.

[39] Sagar Sharma. Epoch vs batch size vs iterations. `https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9`, 2017. (Accessed on 2019/02/20).

[40] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. Technical report, University of Toronto, 2008.

[41] Alexander Strehl, Joydeep Ghosh, and Raymond Mooney. Impact of similarity measures on web-page clustering. Technical report, University of Texas at Austin, 2000.

[42] Chris Tensmeyer and Tony Martinez. Analysis of convolutional neural networks for document image classification. Technical report, 2017.

[43] Alper Kursat Uysal and Serkan Guna. The impact of preprocessing on text classification. Technical report, 2013.

[44] Anish Singh Walia. Activation functions and it's types-which is better? `https://towardsdatascience.com/activation-functions-`

and-its-types-which-is-better-a9a5310cc8f, 2017. (Accessed on 2019/05/20).

[45] Chi-Feng Wang. The vanishing gradient problem. https://towardsdatascience.com/the-vanishing-gradient-problem-69bf08b15484, 2019. (Accessed on 2019/05/07).

[46] Hafidz Zulkifli. Understanding learning rates and how it improves performance in deep learning. https://towardsdatascience.com/understanding-learning-rates-and-how-it-improves-performance-in-deep-learning-d0d4059c1c10, 2018.

[47] Řehůřek et al. Gensim. https://github.com/rare-technologies/gensim. (Accessed on 2019/04/01).