Bachelor Thesis
in Information Systems and Management

Comprehensive Overview of Scene Text Spotting Methods with Deep Learning

Johannes Reichle
Matriculation No. 04797218

Supervisor	Prof. Dr. Rainer Schmidt
Date of Submission	07.03.2022

Declaration

In accordance with §16 para. 10 APO in conjunction with §35 para. 7 RaPO:
I hereby declare that I have written this bachelor thesis independently, have not submitted it for examination purposes elsewhere, have not used any sources or aids other than those indicated, and have marked verbatim and analogous quotations as such.

Munich, the 07.03.2022

. . . .
Johannes Reichle

Abstract

Here abstract for Bachelor Thesis.

Keywords: Deep Learning, Scene Text Spotting, Literature Review

Contents

List of Figures

List of Tables

Abbreviations

AED Average Edit Distance

AP Average Precision

BB Bounding Box

BiLSTM Bidirectional Long Short Term Memory

CNN Convolutional Neural Network

CTC Connectionist Temporal Classification

DL Deep Learning

DNN Deep Neural Network

EnDe Encoder Decoder

FCN Fully Convolutional Neural Network

GD Gradient Descent

IOU Intersection over Union

LSTM Long Short Term Memory

ML Machine Learning

MLP Multi Layer Perceptron

MLS Machine Learning System

MSE Mean Squared Error

NED Normalized Edit Distance

NMS Non Maximum Suppression

NN Neural Network

OCR Optical Character Recognition

RNN Recurrent Neural Network

ROI Region of Interest

RPN Region Proposal Network

SSD Single Shot MultiBox Detector

STD Scene Text Detection

STR Scene Text Recognition

STS Scene Text Spotting

TPS Thin Plate Splines

Notation

Calculus

$\frac{\delta \mathbf{y}}{\delta \mathbf{x}}$ Jacobian matrix $\mathbf{J} \in \mathbb{R}^{n \times m}$ of$f : \mathbb{R}^n \to \mathbb{R}^m$

$\frac{\delta y}{\delta x}$ Partial derivative of y with respect to x

$\frac{dy}{dx}$ Derivative of y with respect to x

$\nabla_{\mathbf{x}} y$ or $\frac{\delta y}{\delta \mathbf{x}}$ Gradient of y with respect to $\mathbf{x}$

Datasets

$\mathbb{X}$ A set of training examples

$\mathbf{x}^{(i)}$ The i-th example (input) from a dataset

$y^{(i)}$ or $\mathbf{y}^{(i)}$ The target associated with $\mathbf{x}^{(i)}$

X The $m \times n$ matrix with input example $\mathbf{x}^{(i)}$ in row $X_{u,:}$

Numbers and Arrays

$\mathbf{a}^T$ or A^T Transposed vector or matrix

A Matrix

a Scalar

$\mathbf{A}$ Tensor

$\mathbf{a}$ Vector

Other

$||\mathbf{x}||$ L^2 norm of $\mathbf{x}$

$\mathbb{R}$ Real numbers

$f(\mathbf{x}; \boldsymbol{\theta})$ A function of $\mathbf{x}$ parametrized by $\boldsymbol{\theta}$

Chapter 1

Introduction

1.1 Motivation

Digitization can be described as transforming analog information into a digital respresentation (Imgrund et al., 2018). Information systems in conjunction with digitization help to optimize efficiency and productivity for business performance, as well as to reduce costs (Imgrund et al., 2018). This facilitates the growing need for automation (Imgrund et al., 2018). Additionally, a comprehensive information system with such digitized information allows a company to aggregate and share information to harness it (Goodhue et al., 1992). However, before reaping the rewards, the information must be digitized in the first place. Take technicians for example, who work in the field with different equipment. It is useful to digitize the lables of such equipment, to keep an overview over the inventory (Abramowicz and Corchuelo, 2019). The automated process of digitizing such data is called Optical Character Recognition (OCR), the concept of extracting typed, handwritten or printed text from an image (Zhao et al., 2020). Techniques for this concept have improved a lot due to the advances in the field of Deep Learning (DL) (Zhao et al., 2020). When compared to traditional methods DL improves automation, effectiveness and generalization (Chen et al., 2021). Applying these new capabilities and finding the right solution in the space of DL for the use case of extracting information of labels is the focus of this thesis. This is an interesting task as performance of OCR systems in natural scenes is still challenging (Zhao et al., 2020; Chen et al., 2021). Such scenes entail natural scenes captured by a camera (Chen et al., 2021; Baek et al., 2019). Factors such as complex backgrounds, noise, perspective and variability in fonts, colors and sizes, of scene texts complicate the process (Hu et al., 2020b; Chen et al., 2021; Baek et al., 2019). In these conditions OCR is known as Scene Text Spotting (STS) (Long et al., 2021).

1.2 Problem Description

The goal of this thesis is to create an overview over possible DL techniques that facilitates finding a solution for the process of digitization. The research question guiding the process is most crucial: Which state of the art DL approaches for scene text OCR are viable for the use case of extracting textual label data from images taken in real world conditions?

The definition of the viability of an approach must be determined for this. What qualities such as detecting alpha-numeric strings or suitability despite inadequate image conditions must a solution have (Ghosh et al., 2017; Hu et al., 2020b)?

It is difficult to assess how well a DL approach performs before it has been implemented and tested on the specific problem or representative dataset (Arpteg et al., 2018). This justifies the need to create an overview rather than pointing out a single approach which is deemed the most promissing. In order to create the overview the necessary steps in the process of STS need to be highlighted, from localizing possible text instances to predicting the characters or words (Long et al., 2021; Sourvanos and Tsatiris, 2018). The ways in which the respective issues for the steps are solved need to be identified from literature, listed and explained alongside.

The article Ashmore et al. (2021) defines four phases of the Machine Learning (ML) lifecycle, namely, Data Management, Model Learing, Model Verification and Model Deployment. Only the substage Model Selection from Model Learning will only be looked at in the scope of this thesis. Goodfellow et al. (2016) states: 'Nearly all deep learning algorithms can be described as particular instances of a fairly simple recipe: combine a specification of a dataset, a cost function, an optimization procedure and a model.' Other aspects such as data analysis, implementation, training, deployment and maintenance of a solution in a production environment shall not be performed. Based on this theses, further further verification, implementation and testing can then be performed. The overview and subsequent analysis thereof creates a foundation for finding the right solution, it does however not contain any claims about the degree of goodness or about the certainty of solving the given problem.

1.3 Methodology

The methodology of this thesis can be labeled as a literature review (Snyder, 2019; Torraco, 2005). The goal is to provide an overview over current DL techniques that can help in choosing which to implement and to test to solve the specific problem of STS, defined in Section 1.2 and more detailed in Chapter 4.

The research question guiding the process is most crucial (Snyder, 2019): Which state of the art DL approaches for STS are viable for the use case of extracting textual label data from images. For a literature review, it is important to report how the information was found and synthesized (Torraco, 2005). Therefore, each section in the overview contains a paragraph about said information. Before getting into current research level, a foundation of knowledge about the field must be layed. A taxonomy is created for this which is useful to classify and give context to innovations in the field. The partition of tasks and categorization of approaches is conducted according to information from overview literature such as Long et al. (2021); Chen et al. (2021); Cong et al. (2019) and difference of approaches that can be identified in research such as Qiao et al. (2021); Sheng et al. (2021); Liu et al. (2020a); Deng et al. (2018). The taxonomy is determined according to the requirement for clarity of subsequent provision of current research.

In order to improve the validity for the subsequent analysis, the problem is dissected further. This includes analysing the specific use case as well as researching which qualities have been identified as generally critical for scene text approaches. The qualities are taken from literature which covers ML in general to literature which covers OCR under challenging scene text conditions.

The strategy for researching current innovations is most important for a literature review (Snyder, 2019). This includes determining databases and keywords that are used, as well as exclusion criteria that are enforced (Torraco, 2005). Innovations are explored through searching in the Google Scholar database. A criterion for further examination is an appropriate amount of citations for the piece of literature in question. Additionally, literature is selected through citations for and by literature which has already been identified as important. All research after 2018 which pertains to extracting scene text is regarded as relevant. Standard OCR solutions may not hold validity in practice, as the image and text conditions can vary in the defined problem (Chen et al., 2021). An important criterion is that the paper contributes to the ML model. This extends to the whole pipeline from extraction features to the final result of the model. The identified literature is synthesized into an overview that is aligned according to the taxonomy.

In the analysis possibly viable approaches are compared with the required qualities defined in Chapter 4. The comparison will be organized according the taxonomy which facilitates the clarity and comprehensibility. The comparison is arranged in hierarchical fashion: different pipeline categories, different categories for pipeline tasks and innovations for those tasks. The analysis thus shows which approaches are worthwhile to apply the whole ML lifecycle to.

1.4 Expected Results

In addition to a deeper understanding of the problem and its detailed definition, the literature review lays the foundation for finding the right approach for the extraction of textual information from images with equipment labels through literature review. In the subsequent analysis different approaches are highlighted for their theoretical fit as a solution.

In the following, the structure of this thesis is listed and each chapter's expected result is detailed along with its benefits for the overall objective of producing an overview of state of the art STS relevant for the problem described in Section 1.2. Chapter 2 lays the theoretical foundation for later chapters. This includes general principles of DL and by extension ML but also of STS. In Chapter 3 current research in regards to the identified requirements is examined. The overview is twofold: a taxonomy for the pipeline and its approaches as well as innovations in current research. The resulting overview can be viewed as a basis for a decision when it comes implementing a practical solution. In Chapter 4 the problem from Section 1.2 is addressed in more detail. The result shall be a firm understanding of qualities that a solution must possess. These requirements are the point of focus for the further examination of techniques. This enables the discussion in Chapter 5. Here not only the results and the availability of a solution but also the methodology of this work is assessed critically. The conclusion is a summary of the results compared to the expected results detailed in this chapter as well as an outlook for further research into the topic.

Chapter 2

Theoretical Foundation

This chapter succinctly describes principles which build the foundation for later chapters. Only the most relevant topics are touched upon, necessary details are explained in later chapters. The mathematics that makes the techniques possible is not explained in depths as it would otherwise exceed the scope of this work. Whenever possible heavy mathematical notation is omitted if it does not aid the understanding of the reader.

2.1 Machine Learning

To grasp DL, a solid understanding of ML has to be developed first (Goodfellow et al., 2016). This is because DL is a subfield of ML (Chauhan and Singh, 2018). The most well known definition for ML comes from Mitchell (1997): 'A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, improves with experience E'.

The task that the Machine Learning System (MLS) learns to perform, can range from approximating a function (e.g. regression — $f : \mathbb{R}^n \to \mathbb{R}^l$, classification — $f : \mathbb{R}^n \to \{1, \ldots, k\}$) to obtaining a different representation for the data that has beneficial properties for further processing but preserves as much information as possible (e.g. PCA for compression) (Goodfellow et al., 2016). Note that the learning itself is not the task but merely the process of improving on performing the task (Goodfellow et al., 2016). One of the most well known ML algorithms is Linear Regression. In the following the algorithm is used as an example for explaining ML principles. As the name implies, Linear Regression is used to predict a value $\hat{y} \in \mathbb{R}$ given the input vector $\mathbf{x} \in \mathbb{R}^n$ which is made up of the features x_i. The goal is to approximate the ground truth y. Linear is derived from the underlying model shown in

Equation 2.1:

$$f(\mathbf{x}; \mathbf{w}, b) = \mathbf{w}^T \cdot \mathbf{x} + b = \sum_{i=1}^{n} w_i x_i + b = \hat{y} \tag{2.1}$$

The scalar product of the weights $\mathbf{w} \in \mathbb{R}^n$ and $\mathbf{x}$ is added to the bias term $b \in \mathbb{R}$. Both $\mathbf{w}, b$ are parameters that are learned by the model in order to optimize the approximation (Goodfellow et al., 2016).

The performance of a model measures how well the task can be completed. Depending on the task of the MLS, different quantitative measures are used. The metric Mean Squared Error (MSE) (see Equation 2.2) can be used for Linear Regression.

$$\mathsf{MSE} = \frac{1}{m} \|(\hat{\mathbf{y}} - \mathbf{y})\|^2 = \frac{1}{m} \sum_{i=1}^{m} \left((\mathbf{w}^T \mathbf{x}^{(i)} + b) - y^{(i)}\right)^2 \tag{2.2}$$

Here m denotes the number of examples $\mathbf{x}^{(i)}$ with the associated targets $y^{(i)}$, used to calculate the error (Géron, 2017; Goodfellow et al., 2016). The goal is to minimize the generalization error which measures the expected performance on previously unseen input (Géron, 2017). For this the test set is used, once the model has been trained. The test set is a part of the available data (Géron, 2017; Goodfellow et al., 2016). The generalization error can be divided into three components. The bias error arises from simplifying assumptions for the model, the variance error measures the variation in the model outcome depending on the data used for training. Both these errors are influenced by the model's representation capacity which is why the relationship between them is call the Bias/Variance tradeoff. Lastly the irreducible error stems from not having measured all data as well as the variation in real data and cannot be reduced (Ashmore et al., 2021; James et al., 2013; Géron, 2017).

The experience part of ML depicts the process where the algorithm is 'experiencing' the training dataset $\mathbb{X}$ and is learning important properties of the dataset. In general, there are two paradigms for training: supervised and unsupervised (Goodfellow et al., 2016). Linear Regression is an example for supervised learning, as the model is using the ground truth value to learn approximating $y^{(i)}$ for the associated input $\mathbf{x}^{(i)}$ (Alzubi et al., 2018; Goodfellow et al., 2016). For unsupervised learning on the other hand the algorithm is not directed to predict a target value but to learn properties about the data and to leverage them for representation tasks like compressing or denoising the data (Goodfellow et al., 2016; Géron, 2017). In most cases training can be described as an optimization problem, i.e. as minimizing a function — the so called objective or loss function L (Goodfellow et al., 2016). The MSE introduced earlier can be used for Linear Regression (see Equation 2.3). This objective function has properties which make it suitable for models which have linear output (Goodfellow et al., 2016).

$$\min_{\mathbf{w},b} \mathsf{MSE}(\mathbf{w}, b) \tag{2.3}$$

Note that for minimization the MSE is a function of $\mathbf{w}, b$ and not of $\mathbf{x}$, in terms of predicting a value the MSE is a function of $\mathbf{x}$ parametrized by $\mathbf{w}, b$ (see Equation 2.3). In Equation 2.1 $\mathbf{w}$,b are parameters that have to be learned in order to minimize the generalization error (James et al., 2013; Géron, 2017). For other tasks such as binary classification, the metric (e.g. F_1-Score) and the objective function (binary cross entropy loss) are different (Géron, 2017; Ho and Wookey, 2020). For optimization the Gradient Descent (GD) algorithm is prevalent, especially in the subfield of DL. As the name suggests, the gradient is used to iteratively update the parameters $\mathbf{w}, b$ to arrive at a minimum of the objective function (see Equation 2.4 and 2.5) (Géron, 2017).

$$\mathbf{w}' \leftarrow \mathbf{w} - \epsilon \cdot \nabla_{\mathbf{w}} \mathsf{MSE}(\mathbf{w}, b) = \mathbf{w} - \frac{2\epsilon}{m} \mathbb{X}^T (\mathbb{X}\mathbf{w} + b - \mathbf{y}) \tag{2.4}$$

$$b' \leftarrow b - \epsilon \cdot \frac{\delta}{\delta b} \mathsf{MSE}(\mathbf{w}, b) = b - \frac{2\epsilon}{m} (\mathbb{X}\mathbf{w} + b - \mathbf{y}) \tag{2.5}$$

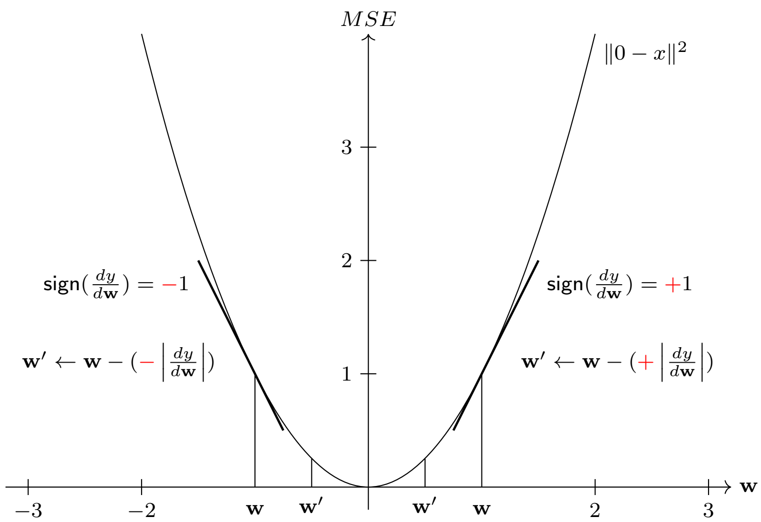
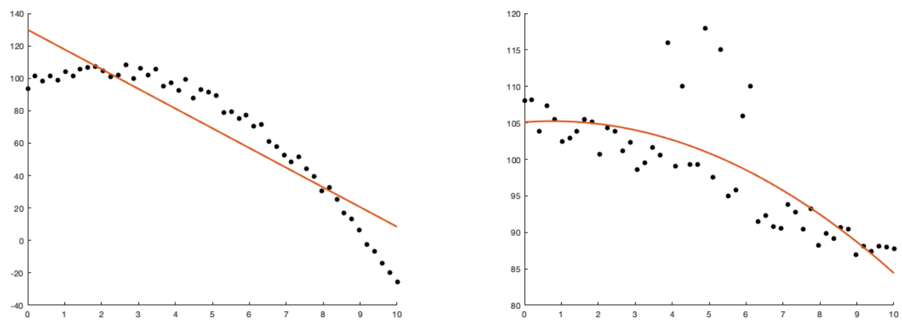


Figure 2.1: Visualization for gradient descent for a 1-dimensional objective function (Goodfellow et al., 2016)

The learning rate constant ϵ can be adjusted to speed up or slow down the 'steps' which can have different effects on the convergence (Goodfellow et al., 2016). There are more sophisticated variations of the GD algorithm which are more suited for practical application (e.g. RMSProp, Adam) (Géron, 2017). Note that the process minimizes the test error with the test set $\mathbb{X}$. The effect on the generalization error depends on model capacity which is the space of functions the model enables (Goodfellow et al., 2016). Linear Regression has the capacity to fit data with a linear relationship between features and ground truth. If the underlying relationship is more complicated, the model can only underfit the data (model bias) (Goodfellow et al., 2016). Polynomial Regression has more capacity for example. Say the real relationship between features and ground truth now actually is linear; the Polynomial Regression model can overfit for statistical outliers in the training set which is why in this case the model with the lower capacity can achieve a lower generalization error (Géron, 2017). Therefore, it is important to improve the bias/variance tradeoff. Aside from model selection, there are different techniques used to prevent overfitting (Regularization) (Goodfellow et al., 2016).



(a) Linear Regression underfit

(b) Polynomial Regression overfit

Figure 2.2: Regression with linear and polynomial model

2.2 Deep Learning

In DL, Deep Neural Networks (DNNs) are leveraged to automatically learn new representations of data through multiple layers of abstraction. This makes DNNs powerful function approximators (Goodfellow et al., 2016). DL has only caught on in the recent years as the big computational cost has been met by improvement in computer hardware as well as in automatic feature learning (Ponti et al., 2017; Chen et al., 2021). In this section the basics of Neural Networks (NNs) are explained and popular basic architectures thereof are introduced.

The most basic NN is called a feedforward NN or Multi Layer Perceptron (MLP) where the information only flows in one direction (in contrast to Recurrent Neural Networks (RNNs) or Transformers) (Goodfellow et al., 2016).

The network is made up of so called artificial neurons. These neurons are arranged as a directed acyclic graph arranged in multiple layers (Goodfellow et al., 2016). The first layer which receives the input features $\mathbf{x}$ is called the input layer, the last layer which outputs the final estimation of $\hat{y}$ or $\hat{\mathbf{y}}$ is called the output layer, all layers in between are called the hidden layers (Shrestha and Mahmood, 2019). The structure with which the NN is build in terms of how many layers, how many neurons in each layer and how they are connected, is called architecture (Goodfellow et al., 2016). The number of layers d is refered to as depths, whereas the dimensionality of those layers is called the width w (Goodfellow et al., 2016). A neuron, the basic building block of NNs,

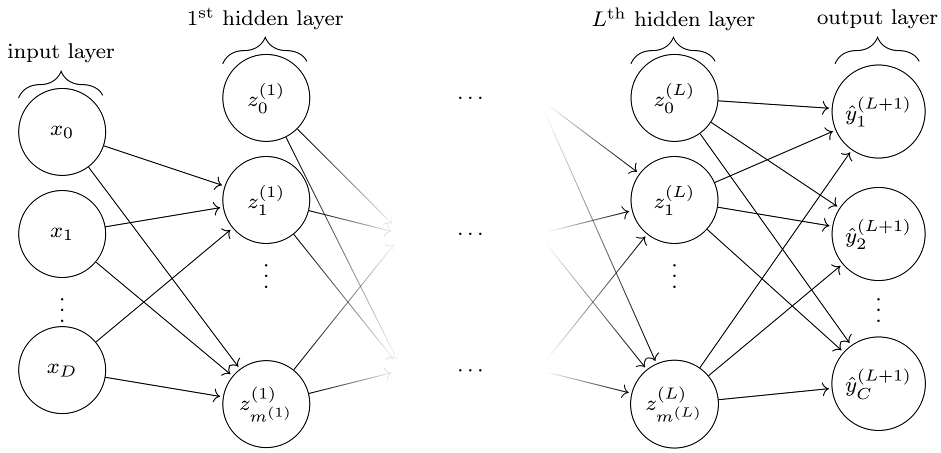


Figure 2.3: Network graph of a $(L+1)$-layer perceptron with D input units and C output units. The l^{th} hidden layer contains $m^{(l)}$ hidden units (Chauhan and Singh, 2018; Goodfellow et al., 2016).

receives input from neurons in the previous layer and calculates a single value which is propagated to neurons in the following layer (Shrestha and Mahmood, 2019). The value is calculated by feeding the received information into a Linear Regression model (see Equation 2.1). The resulting value is fed into an activation function g which introduces nonlinearity, to allow more complicated transformations of information and representation (Goodfellow et al., 2016).

$$f(\mathbf{x};\boldsymbol{\theta}) = g(\boldsymbol{\theta}\mathbf{x}) = \mathbf{z} \tag{2.6}$$

Here f denotes the function which is performed by a layer of neurons (linearity + activation). The parameters of the individual neurons are grouped together to $\boldsymbol{\theta}$ ($\boldsymbol{\theta}_{:,0}$ equals 1 for the bias term). Popular activation functions include ReLU, tanh, sigmoid (σ) and softmax (Shrestha and Mahmood, 2019).

$$\mathsf{ReLU}(x) = \max(0, x) \tag{2.7}$$

$$\tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp x + \exp -x} \tag{2.8}$$

$$\sigma(x) = \frac{1}{1+\exp(-x)} \tag{2.9}$$

$$\mathsf{softmax}(\mathbf{x})_i = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \tag{2.10}$$

While ReLU is the prevalent function for feedforward NN (Goodfellow et al., 2016), tanh is often used in RNNs like in Sherstinsky (2020); Greff et al. (2017). Sigmoid (softmax) activation functions, used for the output layer, are used to generate a bernoulli (multinouli) distribution which is useful for classification tasks (Goodfellow et al., 2016). Note that for e.g. regression, the output layer can omit the activation function (Goodfellow et al., 2016). The calculation of the prediction is basically a concatenation of the functions defined by the layers and their neurons, the process of which is called forwardpropagation (Ponti et al., 2017; Goodfellow et al., 2016).

$$\hat{y} = f(\dots f(f(\mathbf{x};\boldsymbol{\theta}^{(1)});\boldsymbol{\theta}^{(2)})\dots;\boldsymbol{\theta}^{(d)}) \tag{2.11}$$

$\boldsymbol{\theta}^{(i)}$ in Equation 2.11 stands for the parameters in layer i with $\boldsymbol{\theta}^{(i)}_{j,:}$ being the parameters the j-th neuron in that layer (Goodfellow et al., 2016). The forwardpropagation can also be described by a computational graph (see Figure 2.3) (Goodfellow et al., 2016).

The term DNN comes from adding many hidden layers to the NN (Shrestha and Mahmood, 2019). This allows for a more complicated function and better developed features or representations that are extracted from the input feature vector $\mathbf{x}$ (Oyedotun et al., 2015). The DNN can be trained as a whole, thus making feature engineering redundant in contrast to normal ML algorithms (Arpteg et al., 2018). The training algorithm is called backpropagation. The training error is calculated through the objective function and is propagated in conjunction with the output of forwardpropagation on each neuron (Goodfellow et al., 2016). For this the chain rule of calculus can be used to modularly, recursively propagate the loss backwards to use GD (see Figure 2.4). The upstream gradient that is coming from neurons in the next layer is multiplied with the jacobian matrix of the current neuron to produce the downstream gradient that is then used by the preceding layer (Boué, 2018; Goodfellow et al., 2016).

$$\frac{\delta L}{\delta \mathbf{w}} = \frac{\delta L}{\delta \mathbf{z}}\frac{\delta \mathbf{z}}{\delta \mathbf{w}} \tag{2.12}$$

$$\frac{\delta L}{\delta \mathbf{x}} = \frac{\delta L}{\delta \mathbf{z}}\frac{\delta \mathbf{z}}{\delta \mathbf{x}} \tag{2.13}$$

The result of Equation 2.12 is used to update the neuron's weights $\mathbf{w}$ while the result of Equation 2.13 is used for further propagation (Boué, 2018). This calculation is performed until the first layer of the computational graph is

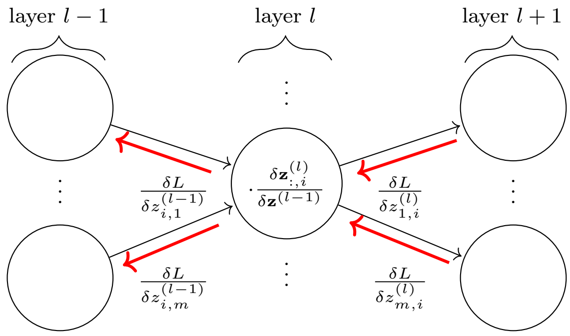


Figure 2.4: Reverse traversing the network's computation graph, $\cdot \frac{\delta \mathbf{z}_{:,i}^{(l)}}{\delta \mathbf{w}_i^{(l)}}$ is used for updating the neurons parameters with gradient descent

reached (Goodfellow et al., 2016). Note that the algorithm can be performed with tensors of arbitrary dimensionality (Goodfellow et al., 2016).

2.3 Convoluational Neural Nets

Convolutional Neural Networks (CNNs) are a type of NN that is also acyclic or feedforward, like MLPs (Chauhan and Singh, 2018). CNNs are specialized to process a grid of values $\mathbf{X}$ like an image (Goodfellow et al., 2016). CNNs are extensively used in computer vision (Chauhan and Singh, 2018). They consist of a variety of components: fully connected layer, activation function, convolutional layer, pooling layer (Chauhan and Singh, 2018; Ponti et al., 2017). The fully connected layers are the layers that make up MLPs (Ponti et al., 2017).

A convolutional layer has multiple filters which consist of multiple kernels (Chauhan and Singh, 2018). For multi layer input, with d so called channels, a filter has the same amount of kernels as there are channels (d) (Ponti et al., 2017). Note that the height and width are refered to as spatial dimensions and the depth is refered to as the channel dimension. A kernel is a $n \times n$ square matrix made up of learnable parameters, so a filter is a tensor $n \times n \times d$. The convolution operation is the elementwise multiplication between the filter and overlapping $n \times n$ subspace of the input (see Figure 2.5) (Ponti et al., 2017). The convolution operation is performed for every space in the input, spaces can be skipped if stride is introduced (Ponti et al., 2017). Often zero-padding is used to preserve the spatial dimensions between input and output of the layer (Ponti et al., 2017). The number of filters a convolutional layer applies is equal to the output channels that the layer has which are often called feature maps (Ponti et al., 2017). The result of the convolution is usually fed into a ReLU activation function to introduce nonlinearity like with

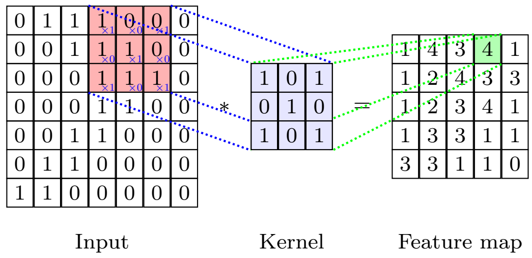


Figure 2.5: Convolution operation of a single kernel (Chauhan and Singh, 2018)

fully connected layers. The activation is performed on every element in the output und preserves the dimensionality (Ponti et al., 2017). In the explained scenario, the filter is slid across a 2d-surface to perform convolution, note that this can be restricted to 1d (for e.g. audio), or extended to 3d (for e.g. CT scans) (Goodfellow et al., 2016).

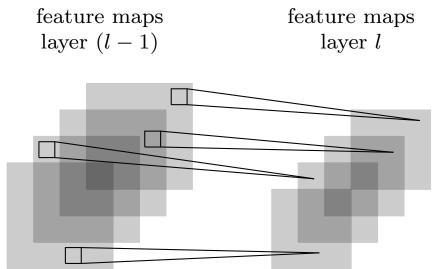


Figure 2.6: Pooling layers preserve channel dimenstions but downsample spatial dimensions

Pooling layers are used to reduce the spatial dimension (i.e. downsampling) of feature maps (Ponti et al., 2017). Pooling layers preserve the number of channels, as the operation is performed to each channel separately (see Figure 2.6) (Chauhan and Singh, 2018). Much like with convolutions (in 2d scenario), the pooling operation is slid across the height and weight dimensions of the channels (possibly with stride) and the pooling operation is performed (Ponti et al., 2017; Chauhan and Singh, 2018). The most popular kind of pooling is maxpooling (Ponti et al., 2017). For the operation only the maximum value of the current subspace of the current channel is taken (see Figure 2.7) (Chauhan and Singh, 2018).

When it comes to deep CNNs, the problem of vanishing/exploding gradients occurs during backpropagation. This problem can be impede conver-

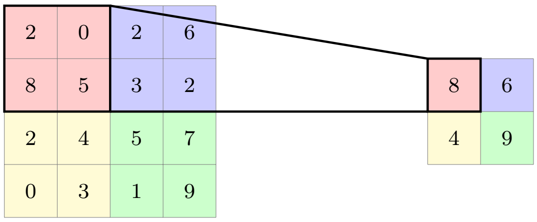


Figure 2.7: 2×2 max pooling operation with stride 2 (Chauhan and Singh, 2018)

gence (He et al., 2015). It can be solved by normalized intitialization of the network's weights and intermediate batch normalization layers (He et al., 2015; Bjorck et al., 2018). Layer normalization can also be used (Liu et al., 2021; Ba et al., 2016). ResNet introduced residual blocks with skip connections which pass the gradient to later layers to bypass exedingly deep NN to overcome the degradation problem (deep NN perform worse than shallow ones) (He et al., 2015). These skip connections basically map the identity of ealier layer to later layers (He et al., 2015).

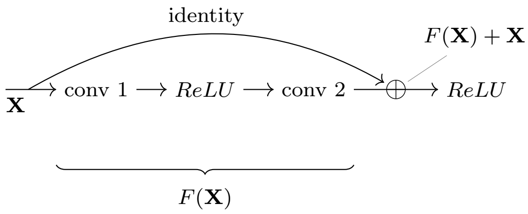


Figure 2.8: Residual block module with skip connection (He et al., 2015)

2.4 Recurrent Neural Nets

RNNs are another popular category of NN which are used for processing sequential input, like text and speech (Chauhan and Singh, 2018). Figure 2.9 shows a simple RNN. The defining element is the recurrent connection from node $\mathbf{h}$ to itself (Goodfellow et al., 2016). A sequence of inputs $\mathbf{X}$ is iteratively fed into the RNN. The current layer of the network takes $\mathbf{x}^{(t)}$ and $\mathbf{h}^{(t-1)}$ and produces $\mathbf{h}^{(t)}$ (see Equation 2.14). Additionally, $\mathbf{h}^{(t)}$ is used to calculate the output $\hat{\mathbf{y}}^{(t)}$ (see Equation 2.15) and it is handed to the next layer (Goodfellow et al., 2016). $\mathbf{h}^{(t)}$ is thought of as a hidden state which stores information from previous inputs (Goodfellow et al., 2016).

$$\mathbf{h}^{(t)} = \tanh(b + W\mathbf{h}^{(t-1)} + U\mathbf{x}^{(t)}) \tag{2.14}$$

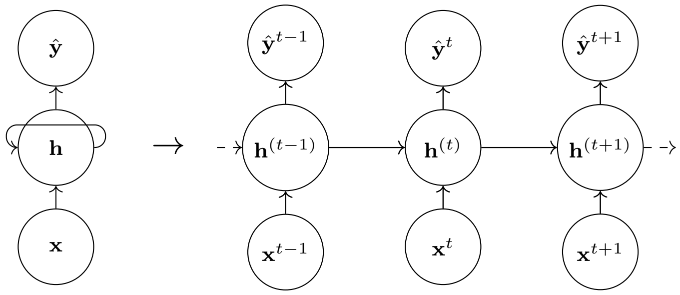


Figure 2.9: Recurrent neural net unrolling (Goodfellow et al., 2016)

$$\hat{\mathbf{y}}^{(t)} = \mathsf{softmax}(c + V\mathbf{h}^{(t)}) \tag{2.15}$$

Note that the connection between hidden layers can differ depending on the type of RNN used (Goodfellow et al., 2016). During an execution run, all unrolled layers share the same weights (U, V, W) (Chauhan and Singh, 2018). This makes gradients from backpropagation vulnerable to exploding/vanishing gradients if the singular values of those weight matrices are > 1 or < 1 (Goodfellow et al., 2016; Pascanu et al., 2013). The example RNN produced a output sequence the same length as the input sequence, however, this does not have to be the case for RNNs (Goodfellow et al., 2016). Both the input and the output can either be a sequence of variable length or a vector of fixed length (Goodfellow et al., 2016). Optimization of the parameters of RNNs is performed with (truncated) backpropagation through time (Sherstinsky, 2020).

The Long Short Term Memory (LSTM) network was introduced by Hochreiter and Schmidhuber (1997) and modified by Gers et al. (1999) to improve longer storing of information from earlier layers (Chauhan and Singh, 2018). The LSTM also improves upon the vanishing gradients problem associated with increasingly deep NNs (Sherstinsky, 2020) (clipping gradients is often used to help with exploding gradients (Goodfellow et al., 2016)). A so called cell is shown in Figure 2.10, it shows the basic structure which represents the recurrent building block of LSTMs (Goodfellow et al., 2016). The three gates (input, forget, output) help make the LSTM a widely used NN model. The gates calculate weights between 0 and 1 (thus the use of σ) that are used to control the flow of information (Goodfellow et al., 2016). The forget gate is part of a self loop which helps to accumulate information longer, the input gate helps to focus on the relevant characteristics of the input information and the output gate removes irrelevant information for the output as well as the

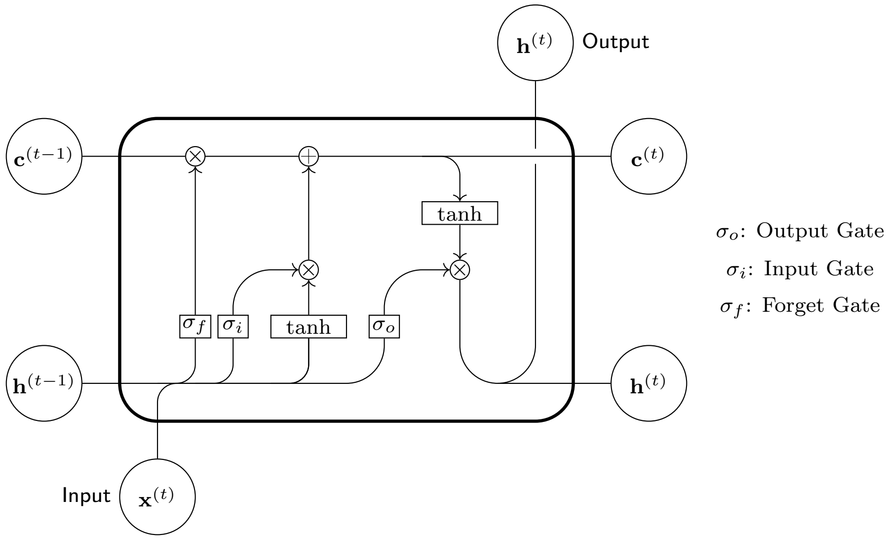


Figure 2.10: Long short term memory cell, merged lines stack vectors, split lines duplicate vector (Goodfellow et al., 2016; Yu et al., 2019)

next cell (Goodfellow et al., 2016).

$$\begin{pmatrix} \mathbf{i}_t \\ \mathbf{f}_t \\ \mathbf{o}_t \\ \mathbf{g}_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} T \begin{pmatrix} \mathbf{x}_t \\ \mathbf{h}_{t-1} \end{pmatrix} \tag{2.16}$$

$$\mathbf{c}_t = \mathbf{f}_t \otimes \mathbf{c}_{t-1} + \mathbf{i}_t \otimes \mathbf{g}_t \tag{2.17}$$

$$\mathbf{h}_t = \mathbf{o}_t \otimes \tanh(\mathbf{c}_t) \tag{2.18}$$

The Equations 2.16, 2.17 and 2.18 show the calculations performed for a single timestep (Xu et al., 2016). σ and tanh denote the application of activation functions after the weight matrix T is multiplied by the stacked input $\mathbf{x}_t, \mathbf{h}_{t-1}$ (Zaremba et al., 2015). $\mathbf{i}_t, \mathbf{f}_t, \mathbf{o}_t$ are the output of the input gate, forget gate and output gate. $\mathbf{c}_t, \mathbf{h}_t$ are the cell states and outputs to the next step or the prediction sequence (Zaremba et al., 2015).

RNNs are often used with the Encoder Decoder (EnDe) mechanism introduced by Cho et al. (2014). This mechanism is a structure for NN that is used for a variety of problems. The EnDe mechanism entails two parts: The encoded which transforms the input into a different representation, it compresses all the information from the input into a feature vector; the decoder uses that the extracted features to generate the output predictions (Asadi and Safabakhsh, 2020; Cho et al., 2014). The decoder can thus use the whole input context at once (Asadi and Safabakhsh, 2020). The output associated with decoders is sequential (Asadi and Safabakhsh, 2020). The mechanism can be implemented in different ways. Both purely RNNs based (Cho et al., 2014) implementations and mixtures along with CNNs (Ghosh et al., 2017) exist (Asadi and Safabakhsh, 2020). The RNN EnDe can be used to transform a sequence of variable length into another sequence with different variable length (Cho et al., 2014). The encoder processes the input sequence and crafts a representation vector that encodes information and context for the whole input sequence which can then be used for every time step in the decoder network (see Figure 2.11) (Cho et al., 2014). An example use for an EnDe architecture

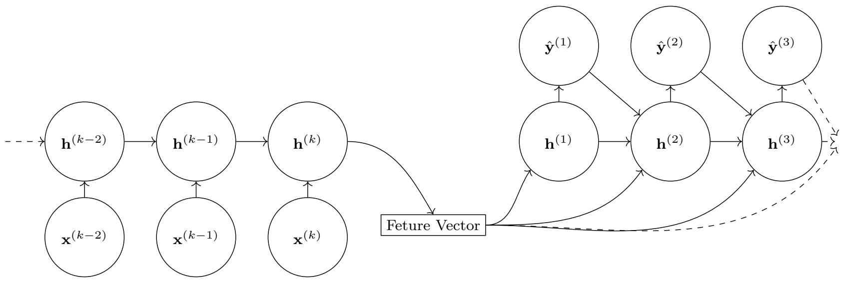


Figure 2.11: Sequence to sequence encoder decoder architecture (Cho et al., 2014)

with a CNN encoder and a RNN decoder would be image captioning (Asadi and Safabakhsh, 2020). The image is encoded into a vector which includes context for the whole image (Asadi and Safabakhsh, 2020).

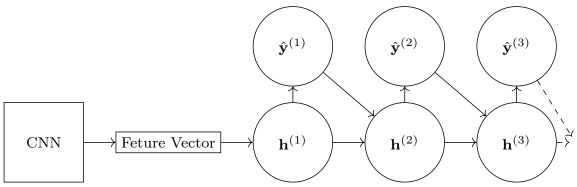


Figure 2.12: Image to sequence encoder decoder architecture (Asadi and Safabakhsh, 2020)

2.5 Scene Text Spotting

OCR is the concept of extracting typed, handwritten or printed text from an image (Zhao et al., 2020). Achieving satisfactory performance of OCR systems in natural scenes is still challenging (Zhao et al., 2020; Chen et al., 2021). Such scenes entail natural scenes captured by a camera (Chen et al., 2021;

Baek et al., 2019). The difficulties arise from diviersity and variability of text, complexity and interference from backgrounds and imperfect imaging conditions. In these conditions OCR is known as STS (Long et al., 2021). Before the advent of DL, researchers in the field had to hand-craft features (Long et al., 2021). DL automates the feature generation process with its representation and learning capabilities (Long et al., 2021; Goodfellow et al., 2016). Because of this, DL methods are the prefered tools for performing STS (Long et al., 2021). OCR and STS are often divided into two subcategories (Scene) Text Detection and (Scene) Text Recognition (Zhao et al., 2020; Long et al., 2021; Chen et al., 2021). For Scene Text Detection (STD) the task is to localize text instances in the image, whereas the Scene Text Recognition (STR) task is to recognize/categorize text from already cropped images (Chen et al., 2021). Note that a system which performs both STR and STD in one continuous pipeline are called end-to-end approaches (Chen et al., 2021).

To assess the performance of the developed approaches, the right evaluation metrics have too be used. The popular protocols Precision, Recall and the F_1-Score are used for comparison among approaches for STD (Long et al., 2021). The metrics are derived with values from the confusion matrix (see Tabel 2.1) (Davis and Goadrich, 2006).

		Ground Truth positive	negative
Prediction	positive	True Positive	False Positive
	negative	False Negative	True Negative

Table 2.1: Confusion Matrix

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \tag{2.19}$$

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \tag{2.20}$$

$$F_1\text{-Score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.21}$$

The difference of metrics for the task manifests itself in the way the values of the confusion matrix are calculated (Long et al., 2021). Note that the tradeoff between False Positives and Fales Negatives manifests itself in the Precision-versus-Recall curve (Su et al., 2015). F_1-Score is also refered to as the harmonic mean (between Precision and Recall) (He et al., 2018b). STD differs mostly in the way the protocols match the prediction to the ground truth (Long et al., 2021). Detectors have multiple predictors which regress the placing and sizing of bounding boxes. More information on this will follow in Chapter 3.

Matching is the process of assigning a bounding box prediction to the ground truth, like in e.g. Liu et al. (2016a); Liao et al. (2018a). The PASCAL approach defines the Intersection over Union (IOU) (see Equation 2.22).

$$IOU = \frac{\text{area of intersection between truth and prediction}}{\text{area of union between truth and prediction}} \tag{2.22}$$

For PASCAL, the prediction will be matched, if the IOU value is larger than a threshhold (Long et al., 2021). A match is considered a True Positive, the other values are assigned accordingly (Sun et al., 2019). Other evaluation approaches are mostly based on IOU, e.g. MSRA-TD 500 evaluates the rotation from the bounding box, compared to the truth in addition to the IOU threshhold (Long et al., 2021). Long et al. (2021) argues that researchers in the field of STD should consider Average Precision (AP) as the main evaluation protocol rather than F_1-Score. According to Su et al. (2015), AP can be considered the area under the Precision-versus-Recall curve. F_1-Score on the other hand only considers singular instances on that curve (Long et al., 2021) and is sensitive to the tradeoff while AP is invariant to it (Shi et al., 2017c). For STR the evaluation can be based on character-level or word-level. There is no need to match ground truth to prediction, as the image is already cropped (Long et al., 2021). Often lexicons which contain possible words, are used by STR. Testing as well as real world performance can depend strongly on these lexicons (Chen et al., 2021; Long et al., 2021). The equations 2.23 and 2.24 show metrics based on word level (Chen et al., 2021).

$$\text{Word Recognition Accuracy} = \frac{\text{correctly recognized words}}{\text{total words}} \tag{2.23}$$

$$\text{Word Error Rate} = 1 - \text{Word Recognition Accuracy} \tag{2.24}$$

An example for a character-based metric would be 1−NED where Normalized Edit Distance (NED) calculates the distance between prediction and ground truth (see Equation 2.25).

$$\text{NED} = \frac{1}{N} \sum_{i=1}^{N} \frac{D(s_i, \hat{s}_i)}{\max(l_i, \hat{l}_i)} \tag{2.25}$$

D denotes the Levenshtein distance, s denotes the text, l denotes the text length and N is the total number of text lines (Shi et al., 2017c). For STR NED is used over the whole dataset (Karatzas et al., 2013). STS is oriented to both STD and STR. The prediction has to be matched to the ground truth like for STD (Long et al., 2021). For comparing predictions and matched the respective ground truths that have been matched, NED is used (Chen et al., 2021). For end-to-end recognition (Karatzas et al., 2013, 2015), the main evaluation protocols that are used include Precision, Recall, F_1-Score and Average Edit Distance (AED) (Chen et al., 2021). A sample is considered a True positive if the NED distance between predictions and matched ground truths is equal to 0 (Sun et al., 2019) (on sample can have multiple text instances). AED is the sum of NED values devided by the number of pictures (Chen et al., 2021). Note that competitions often define their own variants of the metrics, e.g. He et al. (2018b); Shi et al. (2017c). Case sensitivity and matching criteria are examples for changing properties of metrics.

To compare approaches with these metrics benchmark datasets are used which have different characteristics. Table 2.2 lists a couple of influencial

Dataset (year)	**STD**	**STR**	**Text Orientation**	**Characteristics**
ICDAR (2013)	✓	✓	Horizontal	—
IIIT 5K-Word (2012)		✓	Horizontal	Cropped, variance in font, color, size and noise (Long et al., 2021)
ICDAR (2015)	✓	✓	Multi-oriented	Low resolution, small text instances (Liao et al., 2020)
MSRA-TD500 (2012)	✓		Multi-Oriented	Extreme aspect ratios (Liao et al., 2020)
ICDAR MLT (2017)	✓	✓	Curved	Multilingual (Long et al., 2021)
SCUT CTW1500 (2017)	✓		Curved	—
Total-Text (2017)	✓	✓	Curved	—

Table 2.2: Benchmark datasets and their properties

benchmark datasets along with their key properties. ICDAR (2013) references the Focused Scene Text dataset (Karatzas et al., 2013) and ICDAR 2015 to the Incidental Text Competition dataset (Karatzas et al., 2015). The second and third column indicate whether the dataset provides annotations for the tasks. The Text Orientation column specifies the most complicated orientation that is present in the dataset (Curved $\subset$ Multi-oriented $\subset$ Horizontal). A collection of images which shows representational examples taken out of benchmark datasets, can be found in the Appendix A.

Chapter 3

Technique Overview

The objective is to create an overview of techniques in the field. According to the methodology detailed in 1.3, first a taxonomy is introduced which facilitates the analysis and comparison of techniques. The subsequent search for literature which lays the foundation for an overview over current research is documented. Lastly, the advances in the field are placed in the correct position and analyzed.

3.1 Pipeline taxonomy

Before getting into current research level, a base of knowledge about the field has to be layed. A taxonomy is created for this which is useful to classify

Task	Approach category		Identifying properties
STD	Seg free		Localize whole instances
		1-stage	direct BB regression
		2-stage	find ROIs, adjust ROIs for better fit
	Seg based		Localize sub text components to reconstruct instance
		Pixel-level	Pixel level segmentation
		Component-level	Sub-component level segmentation
STR	Seg based		Character segmentation and classification
	EnDe based		Sequence recognition
		CTC based	CTC rule transciption
		Attention based	Attention Mechanism
E2E	2 step		Images are cropped for STR
	2 stage		Features are cropped for STR

Table 3.1: Tasks, method categories and identifying propertis

and give context to innovations in the field (see Table 3.1). The partition of tasks and categorization of approaches is conducted according to overview literature such as Long et al. (2021); Chen et al. (2021); Cong et al. (2019) and related research in the field such as Qiao et al. (2021); Sheng et al. (2021); Liu et al. (2020a); Deng et al. (2018). Note that there can be approaches which blend different categories into one, the taxonomy is merely used to facilitate an overview and to enable a clearer comparison. In order to create the overview the necessary steps in the process of STS need to be highlighted, from localizing possible text instances to predicting the characters or words (Long et al., 2021; Sourvanos and Tsatiris, 2018). Note that details which are not essential are abstracted away. STD and STR only incorporate a part of STS, while end to end approaches incorporate both STD and STR techniques to solve STS (Long et al., 2021; Ghosh et al., 2017; Chen et al., 2021). Therefore this section will first discuss the two parts, to then later combine them.

3.1.1 Scene Text Detection

For STD two main categories of approaches can be identified: segmentation based and Bounding Box (BB) regression based (see Figure 3.1) (Long et al., 2021; Sheng et al., 2021; Liu et al., 2020a). The regression based category draws

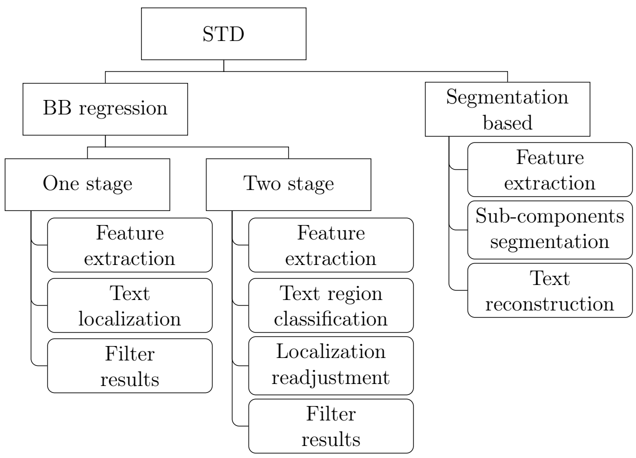


Figure 3.1: Different STD pipelines

heavy inspiration from the field of object detection (Long et al., 2021; Liu et al., 2020a). This is only natural as text detection can be seen as a type of object detection (Liu et al., 2020a; Long et al., 2021). For object detection inspired STD there are two methods: one stage and two stage (Long et al., 2021). Both localize text instances as a whole (in the form of a BB) (Long et al., 2021; Sheng et al., 2021). One stage approaches are modelled after Liu et al. (2016a), Single Shot MultiBox Detector (SSD) and Redmon et al. (2016), You Only Look Once (YOLO). They have in comon that BBs are regressed once and not changed or optimized afterwards (Redmon et al., 2016; Liu et al., 2016a), as opposed to the Region of Interest (ROI) based approach with two stages (Girshick et al., 2014). The basic approach is explained with the example of Liao et al.

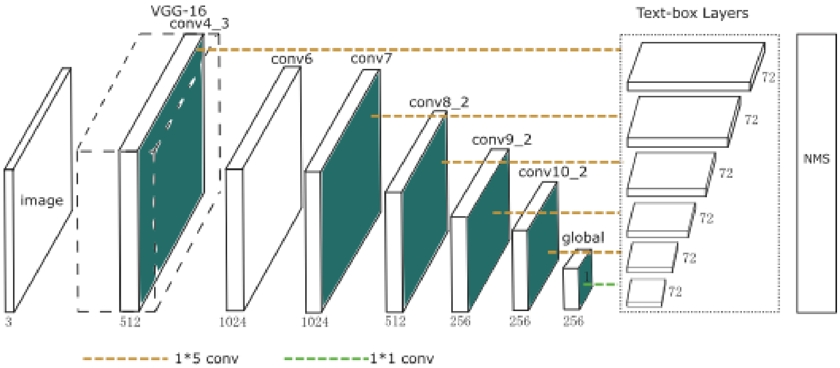


Figure 3.2: Example for a one stage, BB regression based STD architecture (Liao et al., 2017)

(2017) (see Figure 3.2) which is based on SSD. Note that the approach is modeled to recognize horizontal text instances (Liao et al., 2017). It uses convolutional network inspired by the VGG architecture (blocks of: two or three 3×3 conv layers followed by a 2×2 max pooling layer with stride 2) for feature extraction (Liao et al., 2017; Simonyan and Zisserman, 2015). Note that the spatial padding added for convolution ensures that spatial dimensions are preserved (Simonyan and Zisserman, 2015). Afterwards come additional layers which continuously downsample. The output of six of them is separately used as feature maps for BB regression (Liao et al., 2017). The downsampling and BB regression for different layers helps detect text instances of different scales (Liu et al., 2016a). Each spatial location on the feature map can be traced back to a region on the input image (Long et al., 2021). The BB regression is carried out by six convolutional text-box layers which predict how certain (c_1, c_2) the prediction is a text instance or background and where the text instance is (x, y, w, h). Note that the output is not the location of a BB but the offset to the respective anchor box (Liao et al., 2017; Long et al., 2021). Anchor boxes are predefined to give bias towards sizes and aspect ratios of text (Liao et al., 2017). The text-box layers are the difference to the SSD approach for normal object detection (Liao et al., 2017; Liu et al., 2016a). These layers use 1×5 filters to adjust to larger aspect ratios (Liao et al., 2017). Each text-box layer has 72 filters (12 anchor boxes $\cdot$ 6 values per prediction), the filters are slided accross the input features generating 12 predicted BB per position (Liao et al., 2017). The BBs of all layers are then subjected to the process of Non Maximum Suppression (NMS) to filter out the best BB for each possible text instance (Liao et al., 2017). For this NMS is used: of all detections which overlap (IOU) more than a threshhold ϕ only the with the highest confidence score (c) is kept (Hosang et al., 2017).

The R-CNN which builds the foundation for ROI based text detection, was introduced by Girshick et al. (2014) and improved by Girshick (2015) (Fast R-CNN), Ren et al. (2015) (Faster R-CNN) and He et al. (2018a) (Mask R-CNN). Note that 2-stage methods are fully differentiable and thus end to end trainable since Faster R-CNN (like 1-stage methods) (Ren et al., 2015; Long et al., 2021). The two stages consist of: ROI regression, BB adjustments (Jiang et al., 2017; Ren et al., 2015). The two stage STD approach introduced by Jiang et al. (2017) (see Figure 3.3) uses Faster R-CNN.Unlike the previous approach, the architecture is designed to detect multi-oriented text instances (Jiang et al., 2017; Liao et al., 2017). Like with the one stage approach, the two stage

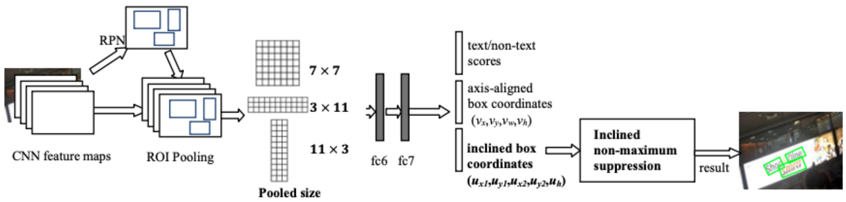


Figure 3.3: Example for a two stage, BB regression based STD architecture (Jiang et al., 2017)

approach starts with feature extraction from the image with a convolutional layers (Jiang et al., 2017). Feature extraction CNN is again inspired by the VGG architecture (Jiang et al., 2017), like in the previous approach. The generated feature map is used by a Region Proposal Network (RPN). Like with the previously explained approach, the feature maps are used to regress the offset respective to bounding boxes, however, these are considered as regions that probably contain text that are subject to change and to be refined in a later stage (Jiang et al., 2017; Lu et al., 2020). The bounding boxes are still axis aligned at this point (Jiang et al., 2017). In contrast to the previous SSD based approach, only one feature map is used in conjunction with BB regression (Jiang et al., 2017). The RPN from Faster R-CNN is adjusted to use smaller scale anchor boxes to adapt to text (Jiang et al., 2017). The resulting BBs are called ROIs (Ren et al., 2015; Jiang et al., 2017). They are used for ROI pooling in conjunction with the original feature maps. This layer uses max pooling to convert the spatial features corresponding to the location of the ROI to a small feature map (Girshick, 2015). In the case of this example, ROI pooling is used to create three feature maps with different aspect ratios ($7 \times 7, 3 \times 11, 11 \times 3$) which are concatenated for the next step (Jiang et al., 2017). The second stage is to predict a confidence score (text, background) for each ROI and to refine them by regressing values (x_1, y_1, x_2, y_2, h) that allow for multi-oriented boxes to account for rotated text (Jiang et al., 2017). At last the resulting BBs are filtered by inclined NMS which is adjusted to the multi-oriented BBs (Jiang et al., 2017).

The basis for the segmentation based methods is the fact that every part of the text instance can be used to verify that there is text (Long et al., 2021). Because of this, sub-text components can be detected separately and then used to re-construct a text instance (Long et al., 2021). Segmentation based methods can roughly be summed up in two categories: pixel level and component level (Long et al., 2021). Like with BB regression based methods, example architectures are explained in order to describe their categories more clearly. The first segmentation based category for STD segments components which are local regions of text that can overlap one or more characters (Long et al., 2021). The architecture (see Figure 3.4) from Shi et al. (2017a) is used

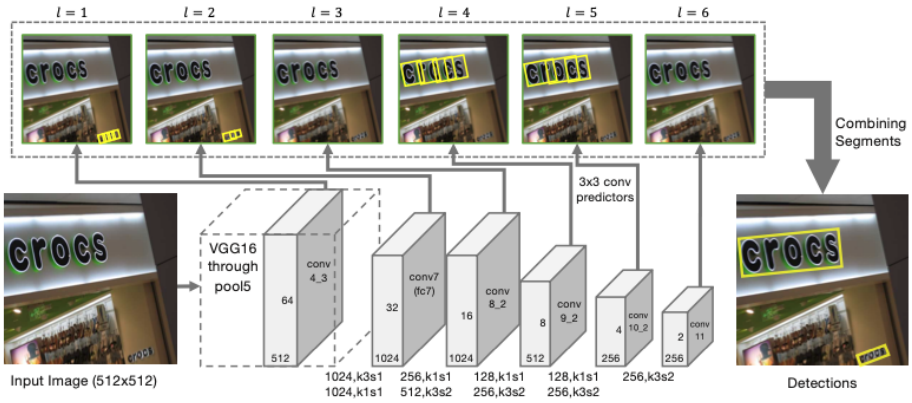


Figure 3.4: Example for a sub-component, segmentation based STD architecture (Shi et al., 2017a)

as an example for this category. Like the one-stage, BB regression based STD approach, the feature extraction CNN of this approach is taken from SSD and thus VGG, the difference is reflected in the prediction layers (Shi et al., 2017a; Liu et al., 2016a; Simonyan and Zisserman, 2015). Instead of detecting whole BBs, the networks predicts both subcomponents and links at multiple scales (Shi et al., 2017a). The convolutional prediction is carried out with seven 3×3 filters followed by a softmax nonlinearity for normalization. The segments are given by the values $x_s, y_s, w_s, h_s, \theta_s$ which offset an anchor box as well as confidence scores c_1, c_2 (Shi et al., 2017a). Links (s_1, s_2) are used to combine the segments and are accordingly used to separate neraby words (Shi et al., 2017a). The convolutional link predictions are then used to as follows: Within layer links are for the neighbors of a space in the predicted feature map (see Figure 3.5 (a), Equation 3.1) (Shi et al., 2017a).

$$\mathcal{N}^{w}_{s^{x,y,l}} = \frac{\{s^{(x',y',l)}\}_{x-1 \le x' \le x+1, y-1 \le y' \le y+1}}{s^{(x,y,l)}} \tag{3.1}$$

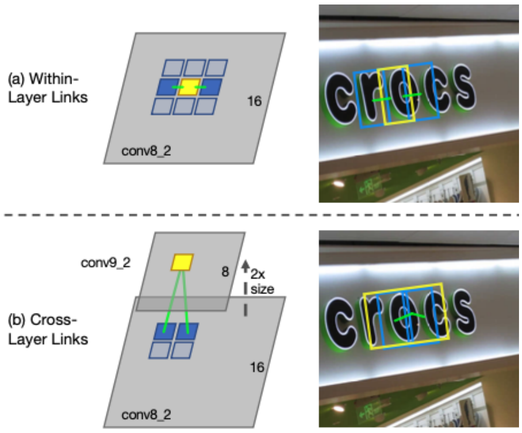


Figure 3.5: Visualization for prediction of links within and cross layers for segmentation based STD (Shi et al., 2017a)

The cross layer links on the other hand are found by using the 4 cross layer neighors of the feature map of the preceding predictor (Shi et al., 2017a). (see Figure 3.5 (b), Equation 3.2) (Shi et al., 2017a).

$$\mathcal{N}^{c}_{s^{x,y,l}} = \{s^{(x',y',l-1)}\}_{2x \leq x' \leq 2x+1, 2y \leq y' \leq 2y+1} \tag{3.2}$$

These cross layer links are used to connect segments on different scales (Shi et al., 2017a). The network architecture designed so that the preceding feature map is twice the size (spatial dimensions) of the current which is necessary to extract the right locations (Shi et al., 2017a). Before reconstruction the text instances, segments are filtered by their confidence scores (Shi et al., 2017a). To reconstruct, the predictions are taken as a graph: segments are nodes, links are edges. Depth-first search is applied to the graph to find connected components and thus text instances (Shi et al., 2017a).

The paper Deng et al. (2018) introduced a pixel level STD approach. It adapts the previous component level STD approach to work on pixel level (Deng et al., 2018). Figure 3.6 shows the CNN structure for feature extraction (until conv5) which is basically VGG architecture with the fully connected layers exchanged with another convolutional stage (Deng et al., 2018). Inspired by Long et al. (2015), the structure then combines downsampled feature maps with later upsampled ones (Deng et al., 2018). Pixel level segmentation approaches mostly rely on Fully Convolutional Neural Networks (FCNs) (Dai et al., 2018). Continuous downsampling and and combining those layers with later, upsampled layers helps to combine coarse, higher level information with fine, lower level information (Long et al., 2015). The upsampling is performed with bilinear interpolation (Deng et al., 2018). The feature extraction is followed by two heads which either predict text/non-text or links (Deng et al.,

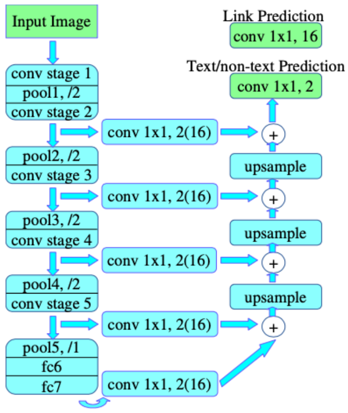


Figure 3.6: PixelNet CNN feature extractor with head structure for pixel segmentation

2018). This creates dense segmentation maps which are used to then segregate different instances of text (Deng et al., 2018). Depending on which head is used, the 1×1 convolution layers either have 2 or $2 \cdot 8$ filters. Counted together, the model has 18 output channels (Deng et al., 2018). 2 filters are used to predict text/non-text for each pixel, while the other 16 filters predict the links (Deng et al., 2018). The text/non-text head essentially performs semantic segmentation, that is to categorize each pixel to its object type (Deng et al., 2018). For every neighbor there is a negative and a positive score. A pixel has eight neighbors: left, left-down, left-up, right, right-down, right-up, up, down. Each of the $2 \cdot 8$ filters is responsible for a neighbor link (Deng et al., 2018). After both links and text/non-text pixels have been predicted, they are

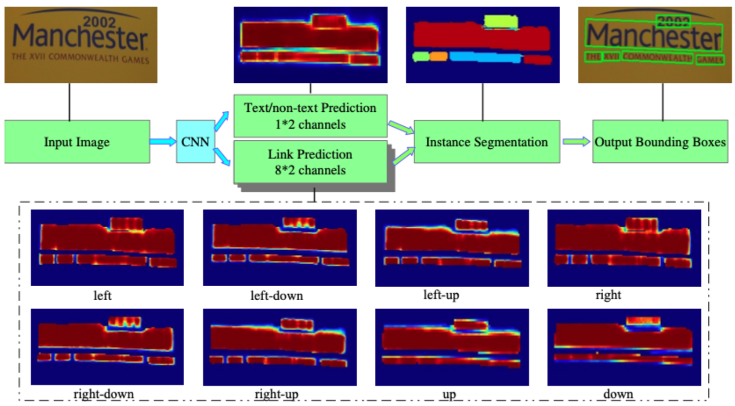


Figure 3.7: Example for a pixel, segmentation based STD architecture (Deng et al., 2018)

combined for instance segmentation (Deng et al., 2018). The link layers are used to indicate whether two text pixels are grouped together an thus belong to the same instance (Deng et al., 2018). The output is a dense prediction map with the same spatial structure as the input image (Deng et al., 2018). A bounding box can then be extracted by laying minimum area rectangles over the instances (Deng et al., 2018). Figure 3.7 shows the whole pipeline.

3.1.2 Scene Text Recognition

For STR two main categories of approaches can be identified: segmentation based and EnDe based (see Figure 3.8) (Chen et al., 2021). Segmentation

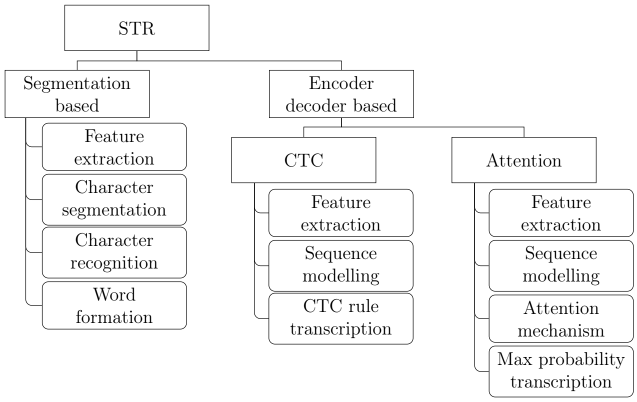


Figure 3.8: Different STR pipelines

based approaches for STR are similar to segmentation based approaches for STD. After feature extraction, the subtext components (characters for STR) are segmented (Chen et al., 2021). Instead of reconstructing a BB, the characters are classified to recognize the text and are then used to form the text instance (Chen et al., 2021). Because of their similarity to STD and the recent domiance of EnDe based approaches (Chen et al., 2021; Long et al., 2021), only the later will be explained in detail with examples. The approach from Wan

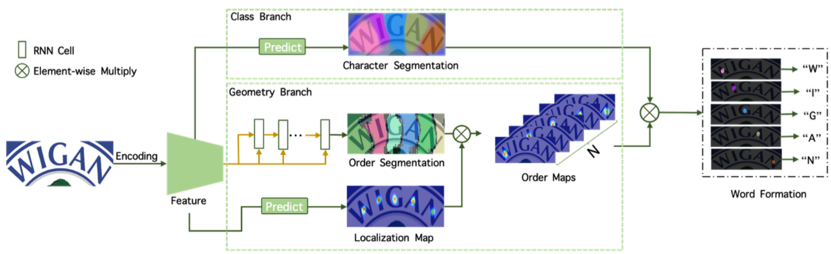


Figure 3.9: Example for segmentation based STR architecture (Wan et al., 2020a)

et al. (2020a) can be used as an example, should the reader feel the need to read up on segmentation based methods for STR that do not include an EnDe mechanism in the prediciton stage. The approach adds a geometry branch parallel to the charcater segmentation which helps with with putting the identified characters in the right sequence (see Figure 3.9) (Wan et al., 2020a).

For EnDe based STR there are twomein categories: Connectionist Temporal Classification (CTC) based and attention based (Chen et al., 2021). These two categories can be distinguished by the way the decoder works (Chen et al., 2021). The previous stages in the pipeline are similar (Long et al., 2021; Chen et al., 2021). Often preprocessing stages are performed to remove distortions, rectify curved text, remove disruptive background, improve resolutions or recover degraded text. Especially rectifying curved text (see Figure 3.10) with the help of thin-plate spines (Bookstein, 1989) and spatial transformer networks (Jaderberg et al., 2015) shows great improvement for STR performance (Long et al., 2021; Chen et al., 2021). For feature extraction different

Figure 3.10: Application of spatial transformer network to rectify curved text (Liu et al., 2016b)

CNN architectures can be chosen. The tradeoff between performance and memory & computation cost determines what CNN should be chosen (Chen et al., 2021). An example for a deep and powerful CNN are ResNets (He et al., 2015) which are often used for benchmark purposes (Chen et al., 2021; Long et al., 2021). Refer to Section 2.3 for more information on ResNets. Efficient feature extraction can e.g. be performed with binary convolutional nets (Liu et al., 2018c). For the sequence modelling and transcription steps first Shi et al. (2017b) (CTC based) and then Ghosh et al. (2017) (attention based) will be used as examples. For Shi et al. (2017b) the sequence is generated by

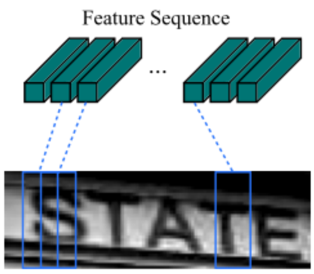


Figure 3.11: Sequential feature extraction from convolution feature maps (Shi et al., 2017b)

taking column out of the feature maps from left to right (see Figure 3.11) (Shi et al., 2017b). This is possible because the parts of the feature map correspond to a spatial region of the original image (Shi et al., 2017b; Goodfellow et al., 2016). The generated sequence features are feed into a deep Bidirectional Long Short Term Memory (BiLSTM). Figure 3.12 shows one layer which is made up of an LSTM which processes the sequence in the correct order and another LSTM which processes the sequence backwards (Shi et al., 2017b). An LSTM processed a sequence to gather context from ealier input for laterr outputs (Shi et al., 2017b; Goodfellow et al., 2016). The BiLSTM allows to gather context of later input for ealier outputs (Shi et al., 2017b). This is important since a character might comprise more than one step in the sequence data (Shi et al., 2017b). The generated encoded features with embedded context are then used

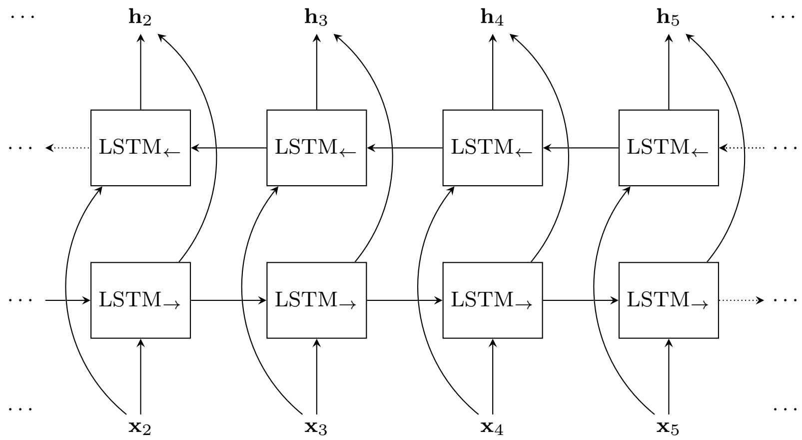


Figure 3.12: Bidirectional LSTM (Goodfellow et al., 2016)

for transcribing the final output (Shi et al., 2017b). The transciption uses conditional probibilities to convert the per-frame features encoded by the deep BiLSTM into the label sequence which is the text instance prediction (Shi et al., 2017b). This process uses the conditional probabilities $p(l\|\mathbf{Y})$ that are defined by Graves et al. (2006), which is why the category of approaches is called CTC based. l stands for the predicted character sequence and $\mathbf{Y}$ stands for the per-frame features (Shi et al., 2017b). Each frame $\mathbf{y}_t$ is a probability distribution over the possible character and a 'blank' character (Shi et al., 2017b; Graves et al., 2006). It is distinguished between lexicon based and lexicon free transcription (Shi et al., 2017b). Lexicon free transcription uses the most probable character or blank (-) at each frame.

HHH-eellll-lll-oo-

Then all duplicate characters are discarded.

H-el-l-o

At last all blanks are discarded, leaving the final prediction (Shi et al., 2017b).

Hello

For lexicon based transcription the combination of $\mathbf{Y}$ is checked for similarity to words in the lexicon based on the levenstein distance (like the metrics AED, NED defined in Section 2.5) (Shi et al., 2017b). The process of searching for similarity can be efficiently implemented with the BK-tree data structure (Burkhard and Keller, 1973; Shi et al., 2017b). The word in the lexicon with the largest probability to be derived from $\mathbf{Y}$ is chosen (Shi et al., 2017b).

For an attention based example, the approach introduced by Ghosh et al. (2017) is used. The approach is inspired by Bahdanau et al. (2016); Xu et al. (2016): It uses a CNN for spatial encoding of the input image which can then be used. The attention mechanism then helps the subsequent LSTM to choose part of the image most important for each timestep (Ghosh et al., 2017). Figure 3.13 shows the network architecture (Ghosh et al., 2017). The

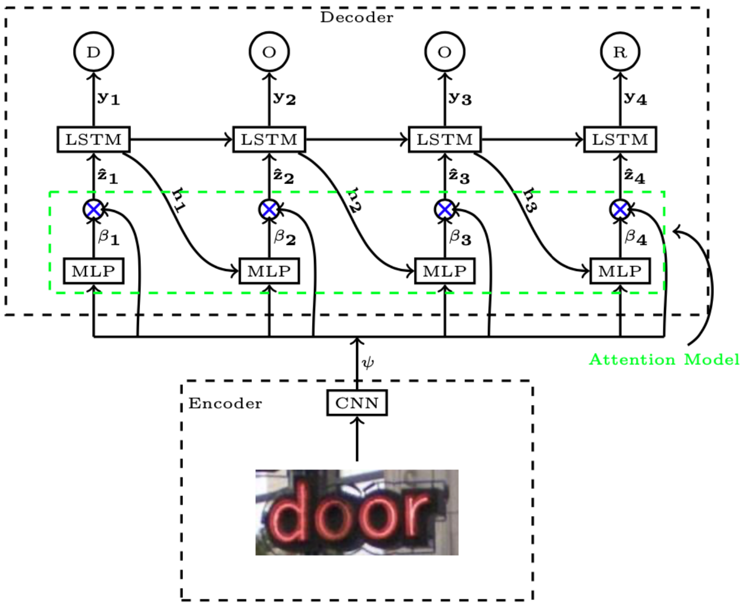


Figure 3.13: Encoder decoder based STR architecture with attention mechanism (Ghosh et al., 2017)

CNN performs feature extraction and encoding (Ghosh et al., 2017). The feature map is then converted into feature vectors like in Figure 3.11 (Ghosh et al., 2017; Shi et al., 2017b). The attention mechanism is placed between the encoder CNN and the decoder LSTM (Ghosh et al., 2017). This mechanism uses a weighted combination of the $\mathsf{softmax}$ activation ($\boldsymbol{\beta}_{t,i}$) of the MLP output (Φ) and the feature vectors to encode the relative importance of the image parts at each time step (see Equations 3.3 and 3.4) (Ghosh et al., 2017; Xu et al., 2016).

$$\hat{\mathbf{z}}_t = \sum_{i=1}^{K} \boldsymbol{\beta}_{t,i} \cdot \mathbf{x}_i \tag{3.3}$$

$$\boldsymbol{\beta}_t = \mathsf{softmax}(\Phi(\mathbf{x}_i, \mathbf{h}_{t-1})) \tag{3.4}$$

Like with the previous approach, the networks output vectors at each time step represent a probability distribution over all possible characters (Ghosh et al., 2017). The transciption process again has the option to be used with a lexicon and without (Ghosh et al., 2017). The basic process without a lexicon leverages the beam search algorithm (Ghosh et al., 2017) to find the sequence $w = [c_1, c_2, \ldots, c_n]$ with the highest probability (Ghosh et al., 2017). For lexicons the beam search algorithm is adjusted so that any sequences that that do not belong to a word in the lexicon fall out of contention (Ghosh et al., 2017). Besides the possibility to use a lexicon, the approach can also incorporate language priors into the beam search process. These priors leverage knowledge about the language that is recognized (Ghosh et al., 2017).

3.1.3 Scene Text Spotting

As mentioned, STD and STR are only parts of the task that is at the center of this thesis. STS can be performed in two ways (see Figure 3.14). (1) Run

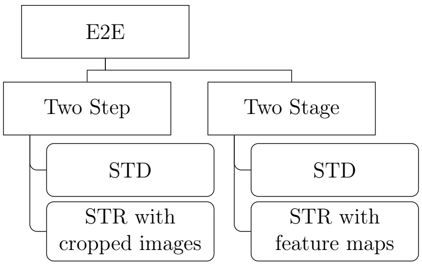


Figure 3.14: Different E2E pipelines

STD, crop the image at the resulting bounding boxes, run STR. (2) Run STD, crop feature maps, run adjusted STR (Chen et al., 2021; Long et al., 2021). Two step methods can modularly be combined (Liao et al., 2017; Shi et al., 2019). The STD approach from Liao et al. (2017) can be combined with the CTC based STR from Shi et al. (2017b). Note that for the two step modular approaches can still be adjusted to better fit together (Liao et al., 2017). The two stage approach that is used as an example was introduced by Busta et al.

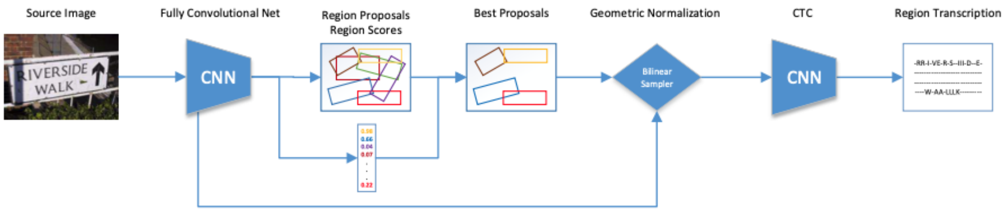


Figure 3.15: 2 stage STS architecture (Busta et al., 2017)

(2017). The approach essentially combines BB based ROI STD with EnDe based CTC STR. After extracting features with a CNN, the approach first generates ROIs. All ROIs with a certainty score above a threshhold are passed over to the recognition module (Busta et al., 2017). The combination of detection and recognition is made possible by bilinear sampling (Busta et al., 2017). This technique helps to collect the correct spatial features in terms of placing, size and rotation to pass to the recognition part (Busta et al., 2017). Note that the transformation normalized the rotation of the ROIs (Busta et al., 2017). At this point ROIs with a confidence score under a threshhold are discarded (Busta et al., 2017). Sampling is the process of selecting the right spatial features from the respective ROI area that can be used for recognition (Liu et al., 2020b). The transformed ROI features are then processed by a CNN which transforms the feature maps into one feature map whose height is defined by the predictable characters and the width is defined by the spatial features of the ROI (Busta et al., 2017). The resulting values can be tranformed into CTC probability distributions (Busta et al., 2017; Graves et al., 2006). Then the CTC transcription rule can be applied to generate the final result. As with normal ROI based approaches, a text instances can be mapped to multiple predictions (Ren et al., 2016; Busta et al., 2017). Instead of performing NMS with the help of the certainty score of ROIs, the filtering is performed with certainty scores which are generated in the recognition phase of the model (Busta et al., 2017).

3.2 State of the Art Methods

This section documents the search for literature which provides the content for the subsequent overview of innovation. For this, important DL techniques and notable advances along with their properties are researched and presented. Note that not the whole approaches that are introduced are explained, but only the innovation that improves upon previous work. For a literature review, it is important to report how the information was found and synthesized (Torraco, 2005). The strategy for researching current research is most important for a literature review (Snyder, 2019). This includes databases and keywords that were used, as well as exclusion criteria that were enforced (Torraco, 2005). The literature is identified through searching in the Google Scholar database. The search is executed with keywords such as, but not only: Scene Text Detection, Scene Text Recognition, Scene Text Spotting, End to End. A criterion for further examination is an appropriate amount of citations (> 100) for the piece of literature in question. All research after 2018 which pertains innovations for model architecture for extracting scene text is regarded as relevant. Standard OCR solutions may not hold validity in practice, as the image and text conditions can vary in the defined problem (Chen et al., 2021). The delimination from Section 1.2 of course holds for this chapter and only literature which concerns advances for the DL model architecture will be regarded as important for the scope of this thesis. This extends to the whole pipeline from extracting features to the final result of the model.

The resulting literature for STD along with the respective innovation can be found in Table 3.2. The conducted search on Google Scholar yielded twelve results until page 10. On page 15 the results started to diverge from the topic and thus the search was concluded. Research like Xue et al. (2018) was not included, since its proposed method concerns the training procedure rather than the model architecture.

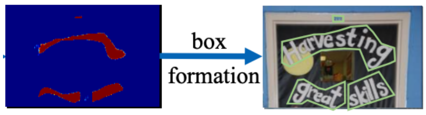
The following approaches are concerned with multi-oriented text: The 1-stage STD approach introduced by Liao et al. (2018c) states that predicting both the text certainty scores and the BBs from the same feature maps does not enable the best performance. This is because regression is dependent on orientation and text classification is not (Liao et al., 2018c). Each prediction has two heads: the regression head and the classification head. The classification head is preceded with oriented response pooling which generates rotation-invariant features from the given feature maps (Liao et al., 2018c). A more efficient approach was introduced by (Liao et al., 2018a). To deal with multi oriented text, the approach introduced a different representation for BBs which is multi-oriented. The represention has 13 parameters (Liao et al., 2018a). That is more than usual for a multi-oriented BB (8) (Ma et al., 2018). The 2-stage approach STD from Ma et al. (2018) deals with text orientation differently. The RPN generates oriented ROIs, they are offset by already rotated anchor boxes (Ma et al., 2018). The results are filtered with NMS which is adjusted to rotation. For the remaining ROIs the ROI pooling operation is also adjusted to deal with rotation, the results of which are used for the final prediction (Ma et al., 2018). The first pixel based appproach was introduced by Deng et al. (2018). It combines dense predictions maps of text/non-text and neighbor links to reconstruct text instances from pixel level (Deng et al., 2018). The approach from Lyu et al. (2018b) is also pixel level segmentation

Approach category		Source	Orien-tation	Innovation	#cit
Seg free					
	1-stage	Liao et al. (2018c)	m	Use rotation sensitive features for regression and rotation insensitive features for text certainty scores	300
		Liao et al. (2018a)	m	Predict multi-oriented bounding boxes with quadrilateral representation	499
	2-stage	Ma et al. (2018)	m	Rotation RPN and rotated ROI pooling	724
Seg based					
	Pixel-level	Deng et al. (2018)	m	Combine text with neighbor predictions for reconstruction	402
		Lyu et al. (2018b)	m	Combine corner localization w position sensitive Segmentation	263
		Liao et al. (2019)	c	Differential binarization for text instance reconstruction	160
		Xu et al. (2019)	c	Segment a direction field — predict directions away from nearest boundary	155
		Wang et al. (2019b)	c	Efficient segmentation and learnable reconstruction	122
		Wang et al. (2019a)	c	Multiple segmentation maps at different scales	266
	Component-level	—			
Mixture					
		Xie et al. (2018)	c	Use text segmentation to rescore ROIs with more contextual information	137
		Dai et al. (2018)	c	Extract finer features, improve NMS with masking	132

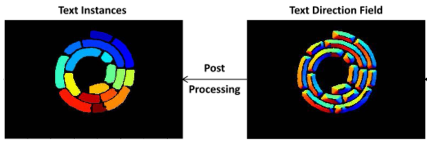
Table 3.2: Notable innovations for STD model architecture;
c:curved, m:multi-oriented, h:horizontal

based. However, not each pixel is predicted separately. The image is subdivided by a grid and each grid space is predicted (Lyu et al., 2018b). Also, text instance corners are predicted. The grid segmentation maps are then used to group corners to reconstruct text instances (Lyu et al., 2018b).

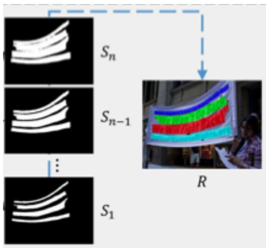
The next approaches introduced innvations regarding curved STD: For Liao et al. (2019), text probability and border pixel-level segmentation maps are used to construct a binary map that represents text/non-text (see Figure 3.16(a)). The construction the binary map is performed with the newly introduced differentiable binarization function (Liao et al., 2019). The binary map can then be used to generate curved BBs (Liao et al., 2019). The approach not only improves performance but also efficiency (Liao et al., 2019). A new representation for curved text was introduced by Xu et al. (2019) (see Figure 3.16(b)). The representation is made up of direction vectors that point away from the nearest border. These vectors are predicted by magnitude and direction (Xu et al., 2019). The resulting text direction field can then be use to reconstruct text instance (Xu et al., 2019). A innovation towards better efficiency based



(a) Binary text segmentation map (Liao et al., 2019)



(b) Textfield with vectors to text center (Xu et al., 2019)



(c) Segmentation scales for label building (Wang et al., 2019a)

Figure 3.16: Different curved text instance reconstructing

on pixel level segmentation from Wang et al. (2019b) introduces a light feature extraction network. The network repeatetly upsamples and downsamples the feature maps to improve the representational capabilities of the efficient extractor (Wang et al., 2019b). The generated features are then used to predict text regions, along with text centers and similarity vectors to distinguish different regions (Wang et al., 2019b). Another way of distinguishing between text instances is to progressively predict larger scale text areas (Wang et al., 2019a) (see Figure 3.16(c)). This is achieved by continuously shrinking the image with the help of kernels (Wang et al., 2019a). This helps as the network can recognize at which scale the instances overlap (Wang et al., 2019a). Recent research has brought up approaches that use both segmentation and BB regression in the same architecture (Xie et al., 2018; Dai et al., 2018). The approach introduced by Xie et al. (2018) leverages text segmentation to rescore certainty for predicted ROIs to improve NMS, the filtered ROIs are then used to refine BBs regression and text classification (Xie et al., 2018). The innovation by Dai et al. (2018) improves feature extraction for STD by fusing lower level features. These features are used for ROI proposals and are then used to pool the fused feates for the text/non-text classification, BB regression and text segmentation branches (Dai et al., 2018). Similar to the previous approach, NMS is modified to use a text segmentation map to better work with curved text instance (Dai et al., 2018; Xie et al., 2018).

For STR the search led to 8 results (see Table 3.3). The last relevant innovation was found on page 13 and at page 18 the search was stopped because the results started to diverge. The only approach (Liao et al., 2018b) that

Approach category		Source	Lexicon	Innovation	#cit
Seg based		Liao et al. (2018b)	w/o	Combine different levels of character attention to localize characters and improve receptive fields with deformable convolution	120
EnDe based					
	CTC based	—			
	Attention based	Zhan and Lu (2019)	b	Iterative rectification with line fitting transformation	181
		Luo et al. (2019)	b	Rectification by removing character offset	174
		Shi et al. (2019)	b	Rectification based on grids	349
		Cheng et al. (2018)	b	Extract and combine character features in four directions	209
		Li et al. (2019)	b	Simple pipeline combined with 2d attention mechanism	141
		Liu et al. (2018a)	b	Adapted attention mechanism to find individual characters and rectify local features for the decoder	112
		Bai et al. (2018)	b	Edit operations (consumption, deletion, insertion) influence probability of prediction	99

Table 3.3: Notable innovations for STR model architecture;
w/: with lexicon, w/o: without lexicon, b: both

is not assigned to attention based actually also contains attention. But the approach works with localizing and classifying each character separately (Liao et al., 2018b) which is why it fits into the segmentation based category. The innovation leverages deformable convolution (Dai et al., 2017) to decide which spatial information to focus on (Liao et al., 2018b). Additionally, it relies on a fully convolutional attention module which weakens background and highlights characters (Liao et al., 2018b). This allows the model to localize each character at pixel level (Liao et al., 2018b). The attention module is performed at different scales (Liao et al., 2018b; Xu et al., 2016). The characters are then classified and formed into a word (Liao et al., 2018b).

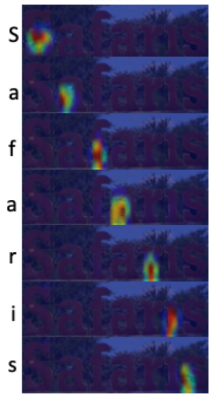
The following approaches are all attention based, however their innovation concern different parts of the attention based STR pipeline. The approaches by Zhan and Lu (2019); Luo et al. (2019); Shi et al. (2019) all improve text rectification ahead of the encoding step. From Zhan and Lu (2019) comes a rectification approach that iteratively estimates the fit of a polynomyal to the text line and uses that prediction to rectify the text with Thin Plate Splines

(TPS) transformation (Bookstein, 1989; Zhan and Lu, 2019). Rectification is improved by Luo et al. (2019) by removing vertical offset of characters to straighten the word out by placing a character lower or higher. This works well for slanted images but has problems with curved text in contrast to the other rectification modules (Zhan and Lu, 2019; Liu et al., 2016b; Long et al., 2021). Fractional pickup is also introduced by Luo et al. (2019). It is a mechanism to improve the attention field of vision to lightly include adjacent characters in order to improve robustness (Luo et al., 2019). Similar to Zhan and Lu (2019), Shi et al. (2019) estimates the text form by localizing grid control-points that surround the text instance from the top and bottom. The grid can then be used in conjunction with TPSs (Shi et al., 2019). The approach by Cheng et al. (2018) improves features by extracting features for text in four directions (right, up, left, down) (Cheng et al., 2018). The feature vectors are then used with the other extracted features to act as a gate (similar to LSTMs) (Cheng et al., 2018). The next two approaches (Li et al., 2019; Liu et al., 2018a) improve upon the attention mechanism for STR. The innovation by Li et al. (2019) uses 2d-attention from (Xu et al., 2016) to use for STR. Additionally, the mechanism is made more robust by taking each of the eight spatial neighbors into account (Li et al., 2019). The mechanism proposed by Liu et al. (2018a) uses attention to recurrently find characters and TPS to rectify them in order predict the text intance. For each character, the corresponding feature region is extracted and is rectified using a local spatial transformer (Liu et al., 2018a). The last relevant STR innovation improves upon the decoding and prediction process. The approach from Bai et al. (2018) introduces the concept of edit probibilities. At each state, the decoder decides which edit operation to perform (consumption, deletion, insertion) based on their edit probibilities (Bai et al., 2018). These operations either forego a time step, add a character to the word or insert a missed character at the previous position (Bai et al., 2018).

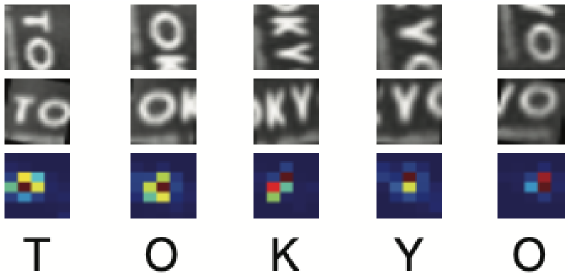
For end to end STS 5 results where identified as relevant under the defined criteria. The last relevant innovation was found on page 5 and at page 10 the topic digerged. The listed two step approaches are not the innovations of the respective methods but rather a a way to extend the proposed STD/STR to a

(a) center line rectification (Zhan and Lu, 2019) (b) offset rectification (Luo et al., 2019) (c) grid rectification (Shi et al., 2019)

Figure 3.17: Visualization of different innovations regarding rectification



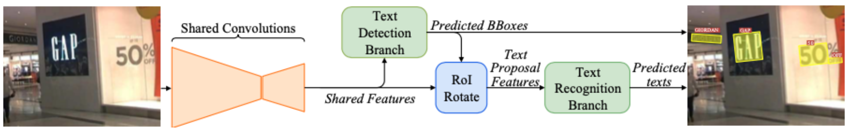
(a) 2d attention (Li et al., 2019)



(b) attention character rectification (Liu et al., 2018a)

Figure 3.18: Effects of innovative attention mechanisms for STR

STS solution (Liao et al., 2018a; Shi et al., 2019). The STD approach from Liao et al. (2018a) can be extended with the CTC based approach from Shi et al. (2017b) which was explained along the taxonomy. The word probability that is predicted by the CTC transcription can be combined with the text certainty score of the STD module to improve NMS. The STR approach from Shi et al. (2019) can be extended with the one stage BB regression based STD approach from Liao et al. (2017) that was also explained in the taxonomy (Shi et al., 2019). The field of end to end STS has moved towards 2 stage approaches



(a) STS architecture with feature sharing (Liu et al., 2018b)

(b) Rotated to axis aligned sampling effect (Liu et al., 2018b)

Figure 3.19: STS architecture with feature sharing and sampling effect (Liu et al., 2018b)

that combin both STD and STR (Lyu et al., 2018a; Long et al., 2021). The innovation from Liu et al. (2018b) combines a oriented detection with combined pixel segmentation and BBs with recognition by using bilinear interpolation

Category	Source	Orien-tation	Lexi-con	Innovation	#cit
2 step	Liao et al. (2018a)	m	b	Combine Texboxes++ detection with CTC recognition, used recognition confidence score for	499
	Shi et al. (2019)	c	b	Textboxes++ BB rectification and grid usage for attention recognition	349
2 stage	Lyu et al. (2018a)	c	w/o	ROI detection followed by pixel and character segmentation	362
	Liu et al. (2018b)	m	b	Bilinear interpolation to transform oriented feature regions into axis-aligned feature maps followed by CTC recognition	355
	Liu et al. (2020b)	c	b	One stage with bezier curves instead of anchor fee detection followed by bezier adjusted ROI pooling and a lightweight CTC recognition head	100

Table 3.4: Notable innovations for E2E model architecture;
c:curved, m:multi-oriented, h:horizontal;
w/: with lexicon, w/o: without lexicon, b: both

to smaple and rotate the spatial features corresponding to a text instance to be axis aligned (Liu et al., 2018b). The subsequent recognition branch is CTC based. The 2 stage approach introduced by Lyu et al. (2018a) uses uses a RPN to propose ROIs which are used to extract features for both a BB regression branch and a segmentation branch which predicts text instances and characters on a pixel level (Lyu et al., 2018a). The BBs are used with the text instance segmentation to form the final detection prediction and the characters are combined to the recognition prediction (Lyu et al., 2018a). The network from Liu et al. (2020b) uses bezier curves for a new representation for curved text. The bezier curve representation is predicted by a one stage, anchor free regression (Liu et al., 2020b). The spatial features corresponding to the predicted instances are then extracted from the feature maps and handed to a lightweight CTC based recognition branch (Lyu et al., 2018a).

Chapter 4

Problem Analysis

This chapter entails an analysis of the problem which is the research question's foundation. It is crucial, as the quality of requirements ultimately determines the quality of the subsequent analysis.

Requirements for a software system that involves ML and thus DL differs from the traditional approach. The data-driven software components are not entirely defined by the programmer but are influenced by data. The system acts with dependency on the test data (Siebert et al., 2021). This poses a challenge in determining requirements and measuring quality of results (Nakamichi et al., 2020). Instead of categorizing functional and non-functional requirements, like for traditional software projects (Zowghi et al., 2014), qualities that a MLS must possess are defined.

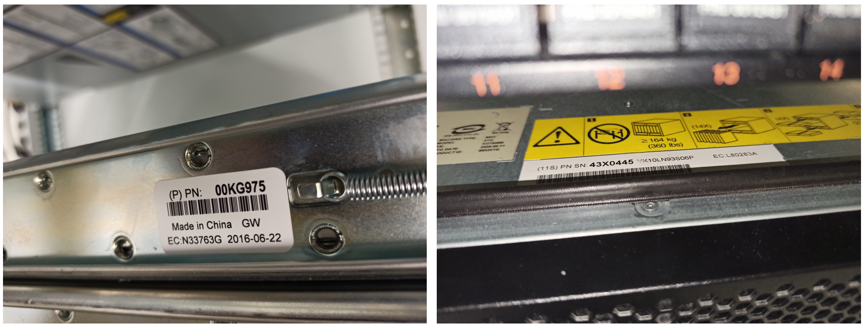
4.1 Use Case

The problem can be depicted by a use case. This use case sets the foundation for determining requirements for an approach because qualities derive from the intended purpose of use (Siebert et al., 2021). Table 4.1 gives an overview over the relevant properties that can be derived from the use case. For this thesis, the basic use case is as follows: A technician takes a photo

Offline Capabilities	Perform extraction process offline
Alphanumeric recognition	Recognize alphanumeric strings such as serial numbers
Semantics retention	Retain semantics given implicitly be space, strucure and rotation of text in labels

Table 4.1: Qualities specific to use case — exclusion criterias

of a device label with his smart phone. For this the technician is situated in locations like a cable shaft. Due to this, there's no internet availability. The process from taking the image to storing the extracted text safely must work offline. The resulting image contains printed textual information which must be extracted by an application on the smart phone. Space and structure of this information can vary from label to label (see figure 4.1). The text does not have to be oriented horizontally. The text, spacing and structure carries semantic information which can be important for later processing in the scope of a business process (Chen et al., 2021). The goal is to extract the text and preserve semantics that are implicitly provided through structure and space. This means text and the respective coordinates, height, width and a possible rotation angle must be output as the result (Yang et al., 2021). Those values can then be transformed into other formats such as JSON or HTML as needed. In addition to this, the labels can contain arbitrary alphanumeric strings such



(a) Positive example (b) Negative example

Figure 4.1: Examples for label images

as serial numbers (see figure 4.1). This results in the requirement that the DL model has to be able to recognize sequences that are not part of a predefined lexicon (Ghosh et al., 2017; Chen et al., 2021). The qualities for the MLS that can be derived directly from the use case (see Table 4.1) can be regarded as excluding criterias, because an approach that does not possess the qualities in question, cannot be regarded as viable for the use case.

4.2 Quality Identification

In the article Ashmore et al. (2021) the qualities are identified and assigned to different challenges in regards to working with MLS: Development Challenges, Production Challenges, Organizational Challenges. Because the only the Model Selection substage of the lifecycle is performed, the challenges and their qualities are not relevant for this thesis, as they concern the operational aspect of MLSs.

In Nakamichi et al. (2020); Siebert et al. (2021) systematic approaches for identification and documentation of qualities are detailed. In MLSs various entities interact to in order to produce the desired functionality. The paper Nakamichi et al. (2020) suggests that in order to adequately evaluate the qualities, it is essential to not only consider the model but the entire MLS (see Figure 4.2). These entities are data, model, environment, sys-

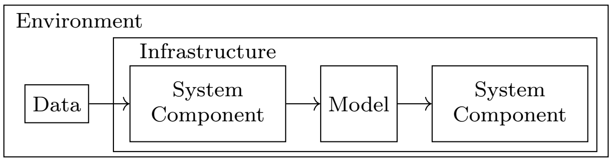


Figure 4.2: Machine learning system entities (Nakamichi et al., 2020)

tem/infrastructure (Nakamichi et al., 2020; Siebert et al., 2021). The article Siebert et al. (2021) differentiates between system and infrastrucure. The infrastrucure represents given hardware and available libraries, whereas the system depicts the software that surrounds the model in the runtime environment. The data view pertains to the quality of development and runtime data (Siebert et al., 2021). The model consists of subcomponents organized in directed acyclc graph building a pipeline. This directed acyclic graph depicts everything from processing the images to the extracted information (Siebert et al., 2021). The environment entity covers the external aspects to the MLS which may interact with it (Siebert et al., 2021). In the scope of this work the environment entails mostly the conditions in which images are taken. For this thesis the entities data and system cannot be regarded as given. The entities environment and infrastrucure are only losely defined through the use case. That is why the systematic approaches cannot be performed in the scope of this thesis. For example Siebert et al. (2021) proposes to follow the systematic CRISP-DM approach of identifying qualities. It cannot be performed due to the lack of data and the other entities derived from the use case.

The Table 4.2 lists all qualities that pertain to the model entity. Different qualities are *grouped together* for their similarities. Because of their properties, they can be evaluated jointly. When it comes to documenting the identified qualities, both Nakamichi et al. (2020) and Siebert et al. (2021) define a meta model for qualities that combines qualities with measurement methods and values and assignes them to an entity of the MLS. The implementation and testing phase are not performed in the scope of this thesis and the difficulty in assessing the performance ahead of those phases, prevents the evaluation of measurements. Additionally, experimental results from literature can only be compared as long as factors such as hardware, platform, source code, configuration and dataset are uniform (Arpteg et al., 2018). Comparing models through results of diferent papers is troublesome, because different papers might use different evaluation and testing environments (Baek et al., 2019). This applies to studies that present an overview such as Chen et al. (2021); Long et al. (2021). These studies can only be regarded as guiding values because the performance for a specific dataset cannot be predicted without testing on it (Arpteg et al., 2018). That's why targets for measurements are not defined, as evaluation would only deliver a false sense of certainty.

Quality	**Sources**
Appropriateness	
Appropriateness	Siebert et al. (2021)
Suitability	Siebert et al. (2021)
Model Fitness — Quality of Output Data	Nakamichi et al. (2020)
Performance	
Performance	Ashmore et al. (2021); Vogelsang and Borg (2019)
Accuracy	Nakamichi et al. (2020)
Model Fitness — Degree of Correctness	Nakamichi et al. (2020); Zhang et al. (2020)
Development correctness	Siebert et al. (2021)
Robustness	
Robustness	Ashmore et al. (2021); Hu et al. (2020a); Siebert et al. (2021)
Robustness Against Change of Input Data	Nakamichi et al. (2020)
Robustness Against Noise Data	Nakamichi et al. (2020)
Relevance / bias-variance tradeoff	Siebert et al. (2021); Zhang et al. (2020)
Trained Model Generalization Performance Appropriateness	Nakamichi et al. (2020)
Reusability	Ashmore et al. (2021)
Interpretability	
Interpretability	Ashmore et al. (2021); Siebert et al. (2021); Zhang et al. (2020)
Understandability	Nakamichi et al. (2020)
Transparency	Arpteg et al. (2018)
Model Explainability	Vogelsang and Borg (2019)
Comprehensibility	Ashmore et al. (2021)
Comprehensiveness	Ashmore et al. (2021)
Fairness	
Fairness	Siebert et al. (2021); Zhang et al. (2020)
Freedom from Discrimination	Vogelsang and Borg (2019)
Performance Efficiency	
Resource Utilization	Siebert et al. (2021); Nakamichi et al. (2020)
Execution efficiency	Siebert et al. (2021)
Temporal Performance	Nakamichi et al. (2020)

Table 4.2: MLS qualities identified for model entity

4.3 Quality Relevancy

In addition to the qualities that arise directly from the use case, literature reveals a number of common qualities in regards to MLS (see Table 4.2), some of which can be regarded as relevant and other do not hold any relevance for the specific use case (see Table 4.3). The qualities are taken from literature which covers ML in general to literature which covers scene text OCR. Only qualities that concern the model will be looked at, as the model is the focus of this thesis. The qualities may however be influenced by other entities.

Relevant	**Irrelevant**
Appropriateness	Fairness
Performance	Interpretability
Robustness	Reusability
Performance efficiency	

Table 4.3: Condensed Qualities for model entity

The appropriateness quality refers to the ability to perform the type of task that is required by the use case (Siebert et al., 2021; Nakamichi et al., 2020). For this thesis, this applies to scene text STS models. Additionally, the properties which are derived from the use case (see Table 4.1), can be grouped under this quality. The offline capabilities are always given in the approaches outlined in Chapter 3. Semantics retention is controlled by the text representation output of the STD model (axis aligned, multi-oriented or quadrilateral bounding box, text/non-text map). Alphanumeric recognition is dependent on how well STR performs without a lexicon.

'An ML model is performant if it operates as expected according to a measure (or set of measures) that captures relevant characteristics of the model output' (Ashmore et al., 2021). For the performance quality, a measure is chosen depending on the type of task to be solved (Siebert et al., 2021). The F-Score is an example for a metric that is used to compare different models Chen et al. (2021); Long et al. (2021). Performance is usually measured with a test dataset that is independent from training and validating a model in order to approximate the generalization performance (Goodfellow et al., 2016; Nakamichi et al., 2020).

The robustness of a model concerns environmental uncertainty (Ashmore et al., 2021). Due to the uncontrolled environment in the practical aspect of taking the images on-site beneficial image properties can not be guaranteed (Chen et al., 2021). Robust text extraction can be influenced by factors such as complex backgrounds, text form (text rotation, font variability, arrangement), image noise (lighting conditions, blur, interference and low resolution) and access (perspective, shape of text) (Oyedotun et al., 2015; Ghosh et al., 2017; Chen et al., 2021). Therefore, these properties have to be accounted for when determining the viability for an approach. Some of these factors do not change the expected prediction (noise), others do (text form)Hu et al. (2020a). An example for bad image quality in regards to STS can be seen in figure 4.1(b). Note that the datasets introduced in Section 2.5 include the challenging image properties, as STS is defined with robustness in mind. For example Karatzas et al. (2013, 2015); Ch'ng and Chan (2017) define their challenge with different image properties concerned with robustness. Additionally, STS is differenciated from OCR by solving more difficult reading problems with more complex image (Long et al., 2021; Hu et al., 2020b; Chen et al., 2021; Baek et al., 2019). Therefore, the difference performance and robustness is not clear cut for STS.

Performance efficiency addresses time and resource utilization when the model is in use. This does not involve the training phase but the execution or prediction (Siebert et al., 2021). The efficiency refers to low latency needs and to minimizing resource needs such as memory usage or power consumption (Nakamichi et al., 2020; Siebert et al., 2021; Sourvanos and Tsatiris, 2018). This quality is especially important for usage on mobile devices in conjunction with DNN (Sourvanos and Tsatiris, 2018; Niu et al., 2019). Note that performance efficiency is heaviliy influenced by the infrastrucure (Nakamichi et al., 2020; Siebert et al., 2021). Because the efficiency needs fall mostly on the model, it is categorized as such and thus deemed relevant in the scope of this thesis.

The first quality often found in research that is not relevant for the use case is fairness. A fair model is free from discrimination bias. For ML this is can be a big problem, since discrimination can not only be influenced through explicit programming in terms of the model but also through implicit knowledge from the data (Vogelsang and Borg, 2019). For the use case however no relevance is attached. The model can either recognize the text or it fails the task.

The interpretability of a model helps to justify the output (Ashmore et al., 2021). The interpretability is twofold: explain what the model has learned, explain how a model given the input comes to the output (Vogelsang and Borg, 2019). This can be challenging for two reasons. ML models used can be complex in terms of size and structure (Ashmore et al., 2021). Modular processing pipelines are continuously replaced with end-to-end models which facilitates the tradeoff between interpretability and performance Arpteg et al. (2018).

Another quality for a ML model refers to how well a model intended for one task can be reused for another related task. This can be beneficial because transfer learning can speed up the training, thus reducing training cost (Ashmore et al., 2021). Reusability is not relavant in the scope of this work as it targets the training phase of the ML lifecycle. The identified relevant qualities will be used in Section 5.1 in order to provide properties to judge the merits of a possible solution.

Chapter 5

Discussion

This chapter contains a comparison between the taxonomy categories that were introduced in Section 3.1. The analysis is followed by a reflection on the methodology and the thesis' results.

5.1 Analysis

The comparison in this section is synthesized from information found in literature as well as some observations regarding recent innovations that are examined in Section 3.2. The analysis is structured similar to the overview sections: first compare approach categories for STD and STR and then move on towards STS. The compared aspects of the approaches are the qualities that were identified as relevant in Section 4.3: appropriateness, performance in conjuntion with robustness, efficiency. Note that appropriateness entails the subqualities that were derived from the use case (see Table 4.1): While the offline capability requirement is met by all of the approaches that are identified in this thesis, semantics retention and alphanumeric recognition have to be analyzed further. STD is the relevant subtask for semantics retention and STR for alphanumeric recognition.

The main challenges for STD involve the tradeoff between speed and accuracy as well as representing arbitrary shaped text instance (Wang et al., 2019b). The innovations regarding STD deal with multi-oriented and curved text (as can be seen in Table 3.2). Segmentation free or BB regression based approaches have trouble with curved text because of the anchor boxes' linear bias (Wang et al., 2019a; Long et al., 2018). Additionally, the regressed BBs have trouble in accurately representing the shape of curved text instances (Long et al., 2021; Wang et al., 2019a). Representing multi-oriented text is not a problem on the other hand (Liao et al., 2018a; Jiang et al., 2017). When it comes to comparing BB regression approaches, one stage approaches are generally more efficient

Approach category		Shortcommings
Seg free		Curved text representation (Long et al., 2021; Wang et al., 2019a) Linear anchor box bias with curved text (Wang et al., 2019a; Long et al., 2018) High aspect ratio test (Shi et al., 2017a; Long et al., 2021)
	One Stage	More efficient, straightforward architecture (Lu et al., 2020)
	Twe Stage	More accurate predictions due to refinement (Lu et al., 2020)
Seg based		Separating different text instances (Wang et al., 2019a) Text instance construction is complex and computation intensive (Xie et al., 2019; Liao et al., 2019; Qiao et al., 2021) Incorporate several computation intensive stages (Dai et al., 2018) More vulnerable to noise (Long et al., 2021)

Table 5.1: STD approach category comparison

because of their straightforward architecture while two stage approaches can generally predict more accurately because of the refinement process (Lu et al., 2020). Long aspect ratios are another problem of scene text which degrades BB regression performance (Shi et al., 2017a; Long et al., 2021). Segmentation based approaches on the other hand take advantage of the fact that every part of the text instance can locally be verified as such (Long et al., 2021). Note that no noteworthy comparison of subcomponent or pixel level segmentation was found. The bottom-up approach alleviates the problem of long aspect ratios (Shi et al., 2017a). However, approach is more computation intensive, as in incorporates more complex stages (Dai et al., 2018) and complicated text instance construction (Xie et al., 2019; Liao et al., 2019; Dai et al., 2018). The text instance construction goes hand in hand with the separation of different instances. This is a challenging task which is vulnerable to noise (Long et al., 2021) and the focus of many new innovations (see Figure 3.16). The segmentation based approaches facilitate more natural representation of curved text (Dai et al., 2018; Long et al., 2021) (see Figure 3.16). What STD is concerned, the bottom line seems to be: Is it curved text detection or efficiency more important?

STR research focuses on accurately recognizing 2d text instance. The innovations found for STR (see Table 3.3) show that the research is mostly focused on robustely recognizing curved text instances (rectification approaches and 2d attention). Additionally, it can be noted that the innovative approaches are mostly attention based. However, attention too has shortcommings (see Figure 5.2). For segmentation based approaches the main shortcommings are as follows. Localizing individual characters is computationally expensive (Zhan and Lu, 2019). A complex pipeline is needed to predict characters and align them (Liu et al., 2020b). Take Wan et al. (2020a) for example which leverages a separate geometry branch to which helps with word formation with the predicted charcters from the classifying branch. Segmentation is also susceptible

Approach category		Shortcommings
Seg based		Accurate detection of individual characters (Chen et al., 2021; Cheng et al., 2018) Disregard for contextual information between characters (Chen et al., 2021) Incorporate several computation intensive stages (Liu et al., 2020b)
EnDe based		Curved and multi-oriented text (Cheng et al., 2018; Long et al., 2021)
	CTC based	Prone to overfitting (Chen et al., 2021) Computation extensive CTC probabilities (Xie et al., 2019) Isolated word recognition (Cong et al., 2019) Not applicable to recognition of 2d text instance (Cheng et al., 2017; Xie et al., 2019; Chen et al., 2021)
	Attention based	Sentence and long sequence recognition (Cong et al., 2019; Chen et al., 2021) Problems without vocabulary (Wan et al., 2020b) Attention drift leads to missing or superfluous characters (Liao et al., 2018b; Xie et al., 2019; Chen et al., 2021) Adapted 2d attention requires a lot of storage and computation (Xie et al., 2019; Chen et al., 2021)

Table 5.2: STR approach category comparison

to errors (Zhan and Lu, 2019; Cheng et al., 2018; Chen et al., 2021). The errors are reinforced by the nature of scene text: complex backgrounds, noise, perspective and other factors (Hu et al., 2020b; Chen et al., 2021; Baek et al., 2019). Additionally, segmentation based approaches cannot make us of contextual information between charcters (Chen et al., 2021). EnDe based STR on the other hand is modeled to take contextual information into account (Long et al., 2021; Chen et al., 2021). The problem manifests itself in dealing with 2d-text instances (Long et al., 2021; Liao et al., 2018b). The EnDe basic approaches use decoders which work with sequences (and therefore 1d) (Long et al., 2021; Cheng et al., 2018). The introduced innovations deal with this in two ways: rectify 2d text into 1d text (Zhan and Lu, 2019; Luo et al., 2019; Shi et al., 2019; Liu et al., 2018a), adapt the attention mechanism to 2d input (Li et al., 2019). Note that CTC cannot be adapted to 2d input (Cheng et al., 2017; Xie et al., 2019) and the attention adaption by Li et al. (2019) is memory and computation intensive (Xie et al., 2019). When comparing CTC to attention based approaches it can be noted that CTC deals better with long text instances or sentences while attention is better with singular words (Cong et al., 2019; Chen et al., 2021). This is because the misalignment in attention can cause missing or superfluous charcters (attention drift) (Bai et al., 2018; Liao et al., 2018b; Cheng et al., 2017). Attention based approaches have the upper hand when it comes to incorporate implicit language models or lexicons but CTC perform better without language priors (Cong et al., 2019). This is especially important because the text in the use case under consideration contains alphanumeric strings which are not part of any lexicon. When it comes to computational efficiency both CTC and attention are expensive and time consuming (Chen et al., 2021).

2 stage approaches have the upper hand, when it comes to end to end STS. These approaches are in the focus for research because they have crucial advantages (see Figure 5.3) (Chen et al., 2021). 2 stage approaches are

Category	Shortcommings
2 step	Error propagation between detection and recognition (Chen et al., 2021; Long et al., 2021) No joint optimization (Qiao et al., 2021; Chen et al., 2021) Computation requirements for two feature extraction CNNs (Liu et al., 2018b; Chen et al., 2021)
2 stage	—

Table 5.3: E2E approach category comparison

more efficient than 2 step approaches because they don't compute feature maps twice (Liu et al., 2018b; Chen et al., 2021). This carries a lot of weight since feature extraction is usually the most time and computation consuming step (Liu et al., 2018b). The direct combination of STD and STR also has impact on the performance. Because of the connecting both stages together are fully differentiable and can be jointly optimized (Chen et al., 2021; Long et al., 2021; Qiao et al., 2021). Without joint optimization error that happen in the detection step can be propagated to the recognition which results in performance degradation (Chen et al., 2021; Qiao et al., 2021).

5.2 Reflection

Chapter 6

Conclusion

Bibliography

Witold Abramowicz and Rafael Corchuelo, editors. *Business Information Systems: 22nd International Conference, BIS 2019, Seville, Spain, June 26–28, 2019, Proceedings, Part II*, volume 354 of *Lecture Notes in Business Information Processing*. Springer International Publishing, Cham, 2019. ISBN 978-3-030-20481-5 978-3-030-20482-2. doi: 10.1007/978-3-030-20482-2. URL http://link.springer.com/10.1007/978-3-030-20482-2.

Jafar Alzubi, Anand Nayyar, and Akshi Kumar. Machine Learning from Theory to Algorithms: An Overview. *Journal of Physics: Conference Series*, 1142:012012, November 2018. ISSN 1742-6588, 1742-6596. doi: 10.1088/1742-6596/1142/1/012012. URL https://iopscience.iop.org/article/10.1088/1742-6596/1142/1/012012.

Anders Arpteg, Björn Brinne, Luka Crnkovic-Friis, and Jan Bosch. Software Engineering Challenges of Deep Learning. In *2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 50–59, August 2018. doi: 10.1109/SEAA.2018.00018.

Ahmad Asadi and Reza Safabakhsh. The Encoder-Decoder Framework and Its Applications. In Witold Pedrycz and Shyi-Ming Chen, editors, *Deep Learning: Concepts and Architectures*, Studies in Computational Intelligence, pages 133–167. Springer International Publishing, Cham, 2020. ISBN 978-3-030-31756-0. doi: 10.1007/978-3-030-31756-0_5. URL https://doi.org/10.1007/978-3-030-31756-0_5.

Rob Ashmore, Radu Calinescu, and Colin Paterson. Assuring the Machine Learning Lifecycle: Desiderata, Methods, and Challenges. *ACM Computing Surveys*, 54(5):1–39, June 2021. ISSN 0360-0300, 1557-7341. doi: 10.1145/3453444. URL https://dl.acm.org/doi/10.1145/3453444.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization. *arXiv:1607.06450 [cs, stat]*, July 2016. URL http://arxiv.org/abs/1607.06450. arXiv: 1607.06450.

Jeonghun Baek, Geewook Kim, Junyeop Lee, Sungrae Park, Dongyoon Han, Sangdoo Yun, Seong Joon Oh, and Hwalsuk Lee. What Is Wrong With Scene Text Recognition Model Comparisons? Dataset and Model Analysis. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4714–4722, Seoul, Korea (South), October 2019. IEEE. ISBN 978-1-72814-803-8. doi: 10.1109/ICCV.2019.00481. URL `https://ieeexplore.ieee.org/document/9010273/`.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv:1409.0473 [cs, stat]*, May 2016. URL `http://arxiv.org/abs/1409.0473`. arXiv: 1409.0473.

Fan Bai, Zhanzhan Cheng, Yi Niu, Shiliang Pu, and Shuigeng Zhou. Edit Probability for Scene Text Recognition. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1508–1516, Salt Lake City, UT, June 2018. IEEE. ISBN 978-1-5386-6420-9. doi: 10.1109/CVPR.2018.00163. URL `https://ieeexplore.ieee.org/document/8578261/`.

Nils Bjorck, Carla P Gomes, Bart Selman, and Kilian Q Weinberger. Understanding Batch Normalization. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL `https://proceedings.neurips.cc/paper/2018/hash/36072923bfc3cf47745d704feb489480-Abstract.html`.

F.L. Bookstein. Principal warps: thin-plate splines and the decomposition of deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(6):567–585, June 1989. ISSN 1939-3539. doi: 10.1109/34.24792. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

Laurent Boué. Deep learning for pedestrians: backpropagation in CNNs. *arXiv:1811.11987 [cs, stat]*, November 2018. URL `http://arxiv.org/abs/1811.11987`. arXiv: 1811.11987.

W. A. Burkhard and R. M. Keller. Some approaches to best-match file searching. *Communications of the ACM*, 16(4):230–236, April 1973. ISSN 0001-0782, 1557-7317. doi: 10.1145/362003.362025. URL `https://dl.acm.org/doi/10.1145/362003.362025`.

Michal Busta, Lukas Neumann, and Jiri Matas. Deep TextSpotter: An End-to-End Trainable Scene Text Localization and Recognition Framework. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages

2223–2231, Venice, October 2017. IEEE. ISBN 978-1-5386-1032-9. doi: 10.1109/ICCV.2017.242. URL http://ieeexplore.ieee.org/document/8237504/.

Nitin Kumar Chauhan and Krishna Singh. A Review on Conventional Machine Learning vs Deep Learning. In *2018 International Conference on Computing, Power and Communication Technologies (GUCON)*, pages 347–352, September 2018. doi: 10.1109/GUCON.2018.8675097.

Xiaoxue Chen, Lianwen Jin, Yuanzhi Zhu, Canjie Luo, and Tianwei Wang. Text Recognition in the Wild: A Survey. *ACM Computing Surveys*, 54(2): 1–35, April 2021. ISSN 0360-0300, 1557-7341. doi: 10.1145/3440756. URL https://dl.acm.org/doi/10.1145/3440756.

Zhanzhan Cheng, Fan Bai, Yunlu Xu, Gang Zheng, Shiliang Pu, and Shuigeng Zhou. Focusing Attention: Towards Accurate Text Recognition in Natural Images. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5086–5094, Venice, October 2017. IEEE. ISBN 978-1-5386-1032-9. doi: 10.1109/ICCV.2017.543. URL http://ieeexplore.ieee.org/document/8237805/.

Zhanzhan Cheng, Yangliu Xu, Fan Bai, Yi Niu, Shiliang Pu, and Shuigeng Zhou. AON: Towards Arbitrarily-Oriented Text Recognition. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5571–5579, Salt Lake City, UT, USA, June 2018. IEEE. ISBN 978-1-5386-6420-9. doi: 10.1109/CVPR.2018.00584. URL https://ieeexplore.ieee.org/document/8578682/.

Chee Kheng Ch'ng and Chee Seng Chan. Total-Text: A Comprehensive Dataset for Scene Text Detection and Recognition. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 01, pages 935–942, November 2017. doi: 10.1109/ICDAR.2017.157. ISSN: 2379-2140.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *arXiv:1406.1078 [cs, stat]*, September 2014. URL http://arxiv.org/abs/1406.1078. arXiv: 1406.1078.

Fuze Cong, Wenping Hu, Qiang Huo, and Li Guo. A Comparative Study of Attention-Based Encoder-Decoder Approaches to Natural Scene Text Recognition. In *2019 International Conference on Document Analysis and Recog-*

nition (ICDAR), pages 916–921, September 2019. doi: 10.1109/ICDAR.2019.00151. ISSN: 2379-2140.

Cong Yao, Xiang Bai, Wenyu Liu, Yi Ma, and Zhuowen Tu. Detecting texts of arbitrary orientations in natural images. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1083–1090, Providence, RI, June 2012. IEEE. ISBN 978-1-4673-1228-8 978-1-4673-1226-4 978-1-4673-1227-1. doi: 10.1109/CVPR.2012.6247787. URL `http://ieeexplore.ieee.org/document/6247787/`.

Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable Convolutional Networks. *arXiv:1703.06211 [cs]*, June 2017. URL `http://arxiv.org/abs/1703.06211`. arXiv: 1703.06211.

Yuchen Dai, Zheng Huang, Yuting Gao, Youxuan Xu, Kai Chen, Jie Guo, and Weidong Qiu. Fused Text Segmentation Networks for Multi-oriented Scene Text Detection. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 3604–3609, August 2018. doi: 10.1109/ICPR.2018.8546066. ISSN: 1051-4651.

Jesse Davis and Mark Goadrich. The relationship between Precision-Recall and ROC curves. In *Proceedings of the 23rd international conference on Machine learning - ICML '06*, pages 233–240, Pittsburgh, Pennsylvania, 2006. ACM Press. ISBN 978-1-59593-383-6. doi: 10.1145/1143844.1143874. URL `http://portal.acm.org/citation.cfm?doid=1143844.1143874`.

Dan Deng, Haifeng Liu, Xuelong Li, and Deng Cai. PixelLink: Detecting Scene Text via Instance Segmentation. *arXiv:1801.01315 [cs]*, January 2018. URL `http://arxiv.org/abs/1801.01315`. arXiv: 1801.01315.

Felix A. Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to Forget: Continual Prediction with LSTM. *Neural Computation*, 12:2451–2471, 1999.

Suman K. Ghosh, Ernest Valveny, and Andrew D. Bagdanov. Visual Attention Models for Scene Text Recognition. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 01, pages 943–948, November 2017. doi: 10.1109/ICDAR.2017.158. ISSN: 2379-2140.

Ross Girshick. Fast R-CNN. *arXiv:1504.08083 [cs]*, September 2015. URL `http://arxiv.org/abs/1504.08083`. arXiv: 1504.08083.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *arXiv:1311.2524 [cs]*, October 2014. URL `http://arxiv.org/abs/1311.2524`. arXiv: 1311.2524.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. Adaptive computation and machine learning. The MIT Press, Cambridge, Massachusetts, 2016. ISBN 978-0-262-03561-3.

Dale L. Goodhue, Michael D. Wybo, and Laurie J. Kirsch. The Impact of Data Integration on the Costs and Benefits of Information Systems. *MIS Quarterly*, 16(3):293, September 1992. ISSN 02767783. doi: 10.2307/249530. URL `https://www.jstor.org/stable/249530?origin=crossref`.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 369–376, New York, NY, USA, June 2006. Association for Computing Machinery. ISBN 978-1-59593-383-6. doi: 10.1145/1143844.1143891. URL `https://doi.org/10.1145/1143844.1143891`.

Klaus Greff, Rupesh Kumar Srivastava, Jan Koutník, Bas R. Steunebrink, and Jürgen Schmidhuber. LSTM: A Search Space Odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28(10):2222–2232, October 2017. ISSN 2162-237X, 2162-2388. doi: 10.1109/TNNLS.2016.2582924. URL `http://arxiv.org/abs/1503.04069`. arXiv: 1503.04069.

Aurélien Géron. *Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, Beijing ; Boston, first edition edition, 2017. ISBN 978-1-4919-6229-9. OCLC: ocn953432302.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *arXiv:1512.03385 [cs]*, December 2015. URL `http://arxiv.org/abs/1512.03385`. arXiv: 1512.03385.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. *arXiv:1703.06870 [cs]*, January 2018a. URL `http://arxiv.org/abs/1703.06870`. arXiv: 1703.06870.

Mengchao He, Yuliang Liu, Zhibo Yang, Sheng Zhang, Canjie Luo, Feiyu Gao, Qi Zheng, Yongpan Wang, Xin Zhang, and Lianwen Jin. ICPR2018 Contest on Robust Reading for Multi-Type Web Images. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 7–12, August 2018b. doi: 10.1109/ICPR.2018.8546143. ISSN: 1051-4651.

Yaoshiang Ho and Samuel Wookey. The Real-World-Weight Cross-Entropy Loss Function: Modeling the Costs of Mislabeling. *IEEE Access*, 8:4806–4813, 2020. ISSN 2169-3536. doi: 10.1109/ACCESS.2019.2962617. Conference Name: IEEE Access.

Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. Conference Name: Neural Computation.

Jan Hosang, Rodrigo Benenson, and Bernt Schiele. Learning Non-maximum Suppression. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6469–6477, Honolulu, HI, July 2017. IEEE. ISBN 978-1-5386-0457-1. doi: 10.1109/CVPR.2017.685. URL `http://ieeexplore.ieee.org/document/8100168/`.

Boyue Caroline Hu, Rick Salay, Krzysztof Czarnecki, Mona Rahimi, Gehan Selim, and Marsha Chechik. Towards Requirements Specification for Machine-learned Perception Based on Human Performance. In *2020 IEEE Seventh International Workshop on Artificial Intelligence for Requirements Engineering (AIRE)*, pages 48–51, September 2020a. doi: 10.1109/AIRE51212.2020.00014.

Wenyang Hu, Xiaocong Cai, Jun Hou, Shuai Yi, and Zhiping Lin. GTC: Guided Training of CTC Towards Efficient and Accurate Scene Text Recognition. *arXiv:2002.01276 [cs, eess]*, February 2020b. URL `http://arxiv.org/abs/2002.01276`. arXiv: 2002.01276.

Florian Imgrund, Marcus Fischer, Christian Janiesch, and Axel Winkelmann. *Approaching Digitalization with Business Process Management*. March 2018.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, and koray kavukcuoglu. Spatial Transformer Networks. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL `https://proceedings.neurips.cc/paper/2015/hash/33ceb07bf4eeb3da587e268d663aba1a-Abstract.html`.

Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani, editors. *An introduction to statistical learning: with applications in R*. Number 103 in Springer texts in statistics. Springer, New York, 2013. ISBN 978-1-4614-7137-0. OCLC: ocn828488009.

Yingying Jiang, Xiangyu Zhu, Xiaobing Wang, Shuli Yang, Wei Li, Hua Wang, Pei Fu, and Zhenbo Luo. R2CNN: Rotational Region CNN for Orientation

Robust Scene Text Detection. *arXiv:1706.09579 [cs]*, June 2017. URL `http://arxiv.org/abs/1706.09579`. arXiv: 1706.09579.

Dimosthenis Karatzas, Faisal Shafait, Seiichi Uchida, Masakazu Iwamura, Lluis Gomez i Bigorda, Sergi Robles Mestre, Joan Mas, David Fernandez Mota, Jon Almazàn Almazàn, and Lluís Pere de las Heras. ICDAR 2013 Robust Reading Competition. In *2013 12th International Conference on Document Analysis and Recognition*, pages 1484–1493, August 2013. doi: 10.1109/ICDAR.2013.221. ISSN: 2379-2140.

Dimosthenis Karatzas, Lluis Gomez-Bigorda, Anguelos Nicolaou, Suman Ghosh, Andrew Bagdanov, Masakazu Iwamura, Jiri Matas, Lukas Neumann, Vijay Ramaseshan Chandrasekhar, Shijian Lu, Faisal Shafait, Seiichi Uchida, and Ernest Valveny. ICDAR 2015 competition on Robust Reading. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 1156–1160, August 2015. doi: 10.1109/ICDAR.2015.7333942.

Hui Li, Peng Wang, Chunhua Shen, and Guyu Zhang. Show, Attend and Read: A Simple and Strong Baseline for Irregular Text Recognition. *arXiv:1811.00751 [cs]*, March 2019. URL `http://arxiv.org/abs/1811.00751`. arXiv: 1811.00751.

Minghui Liao, Baoguang Shi, Xiang Bai, Xinggang Wang, and Wenyu Liu. TextBoxes: A Fast Text Detector with a Single Deep Neural Network. In *Thirty-First AAAI Conference on Artificial Intelligence*, February 2017. URL `https://www.aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14202`.

Minghui Liao, Baoguang Shi, and Xiang Bai. TextBoxes++: A Single-Shot Oriented Scene Text Detector. *IEEE Transactions on Image Processing*, 27 (8):3676–3690, August 2018a. ISSN 1057-7149, 1941-0042. doi: 10.1109/TIP.2018.2825107. URL `http://arxiv.org/abs/1801.02765`. arXiv: 1801.02765.

Minghui Liao, Jian Zhang, Zhaoyi Wan, Fengming Xie, Jiajun Liang, Pengyuan Lyu, Cong Yao, and Xiang Bai. Scene Text Recognition from Two-Dimensional Perspective. *arXiv:1809.06508 [cs]*, November 2018b. URL `http://arxiv.org/abs/1809.06508`. arXiv: 1809.06508.

Minghui Liao, Zhen Zhu, Baoguang Shi, Gui-song Xia, and Xiang Bai. Rotation-Sensitive Regression for Oriented Scene Text Detection. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages

5909–5918, Salt Lake City, UT, USA, June 2018c. IEEE. ISBN 978-1-5386-6420-9. doi: 10.1109/CVPR.2018.00619. URL `https://ieeexplore.ieee.org/document/8578717/`.

Minghui Liao, Zhaoyi Wan, Cong Yao, Kai Chen, and Xiang Bai. Real-time Scene Text Detection with Differentiable Binarization. *arXiv:1911.08947 [cs]*, December 2019. URL `http://arxiv.org/abs/1911.08947`. arXiv: 1911.08947.

Minghui Liao, Guan Pang, Jing Huang, Tal Hassner, and Xiang Bai. Mask TextSpotter v3: Segmentation Proposal Network for Robust Scene Text Spotting. *arXiv:2007.09482 [cs]*, July 2020. URL `http://arxiv.org/abs/2007.09482`. arXiv: 2007.09482.

Fenglin Liu, Xuancheng Ren, Zhiyuan Zhang, Xu Sun, and Yuexian Zou. Rethinking Skip Connection with Layer Normalization in Transformers and ResNets. *arXiv:2105.07205 [cs]*, May 2021. URL `http://arxiv.org/abs/2105.07205`. arXiv: 2105.07205.

Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single Shot MultiBox Detector. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, Lecture Notes in Computer Science, pages 21–37, Cham, 2016a. Springer International Publishing. ISBN 978-3-319-46448-0. doi: 10.1007/978-3-319-46448-0_2.

Wei Liu, Chaofeng Chen, Kwan-YeeK. Wong, Zhizhong Su, and Junyu Han. STAR-Net: A SpaTial Attention Residue Network for Scene Text Recognition. In *Procedings of the British Machine Vision Conference 2016*, pages 43.1–43.13, York, UK, 2016b. British Machine Vision Association. ISBN 978-1-901725-59-9. doi: 10.5244/C.30.43. URL `http://www.bmva.org/bmvc/2016/papers/paper043/index.html`.

Wei Liu, Chaofeng Chen, and Kwan-Yee Wong. Char-Net: A Character-Aware Neural Network for Distorted Scene Text Recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), April 2018a. ISSN 2374-3468. URL `https://ojs.aaai.org/index.php/AAAI/article/view/12246`. Number: 1.

Xi Liu, Gaojing Zhou, Rui Zhang, and Xiaolin Wei. An Accurate Segmentation-Based Scene Text Detector with Context Attention and Repulsive Text Border. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2344–2352, Seattle, WA, USA, June 2020a. IEEE. ISBN 978-1-72819-360-1. doi: 10.1109/CVPRW50498.2020.00283. URL `https://ieeexplore.ieee.org/document/9151062/`.

Xuebo Liu, Ding Liang, Shi Yan, Dagui Chen, Yu Qiao, and Junjie Yan. FOTS: Fast Oriented Text Spotting with a Unified Network. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5676–5685, Salt Lake City, UT, USA, June 2018b. IEEE. ISBN 978-1-5386-6420-9. doi: 10.1109/CVPR.2018.00595. URL `https://ieeexplore.ieee.org/document/8578693/`.

Yuliang Liu, Hao Chen, Chunhua Shen, Tong He, Lianwen Jin, and Liangwei Wang. ABCNet: Real-Time Scene Text Spotting With Adaptive Bezier-Curve Network. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9806–9815, Seattle, WA, USA, June 2020b. IEEE. ISBN 978-1-72817-168-5. doi: 10.1109/CVPR42600.2020.00983. URL `https://ieeexplore.ieee.org/document/9156344/`.

Zichuan Liu, Yixing Li, Fengbo Ren, Wang Ling Goh, and Hao Yu. Squeezed-Text: a real-time scene text recognition by binary convolutional encoder-decoder network. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18, pages 7194–7201, New Orleans, Louisiana, USA, February 2018c. AAAI Press. ISBN 978-1-57735-800-8.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully Convolutional Networks for Semantic Segmentation. *arXiv:1411.4038 [cs]*, March 2015. URL `http://arxiv.org/abs/1411.4038`. arXiv: 1411.4038.

Shangbang Long, Jiaqiang Ruan, Wenjie Zhang, Xin He, Wenhao Wu, and Cong Yao. TextSnake: A Flexible Representation for Detecting Text of Arbitrary Shapes. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, volume 11206, pages 19–35. Springer International Publishing, Cham, 2018. ISBN 978-3-030-01215-1 978-3-030-01216-8. doi: 10.1007/978-3-030-01216-8_2. URL `http://link.springer.com/10.1007/978-3-030-01216-8_2`. Series Title: Lecture Notes in Computer Science.

Shangbang Long, Xin He, and Cong Yao. Scene Text Detection and Recognition: The Deep Learning Era. *International Journal of Computer Vision*, 129(1):161–184, January 2021. ISSN 0920-5691, 1573-1405. doi:

10.1007/s11263-020-01369-0. URL https://link.springer.com/10.1007/s11263-020-01369-0.

Xin Lu, Quanquan Li, Buyu Li, and Junjie Yan. MimicDet: Bridging the Gap Between One-Stage and Two-Stage Object Detection. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, Lecture Notes in Computer Science, pages 541–557, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58568-6. doi: 10.1007/978-3-030-58568-6_32.

Canjie Luo, Lianwen Jin, and Zenghui Sun. A Multi-Object Rectified Attention Network for Scene Text Recognition. *arXiv:1901.03003 [cs]*, January 2019. URL http://arxiv.org/abs/1901.03003. arXiv: 1901.03003.

Pengyuan Lyu, Minghui Liao, Cong Yao, Wenhao Wu, and Xiang Bai. Mask TextSpotter: An End-to-End Trainable Neural Network for Spotting Text with Arbitrary Shapes. pages 67–83, 2018a. URL https://openaccess.thecvf.com/content_ECCV_2018/html/Pengyuan_Lyu_Mask_TextSpotter_An_ECCV_2018_paper.html.

Pengyuan Lyu, Cong Yao, Wenhao Wu, Shuicheng Yan, and Xiang Bai. Multi-oriented Scene Text Detection via Corner Localization and Region Segmentation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7553–7563, Salt Lake City, UT, USA, June 2018b. IEEE. ISBN 978-1-5386-6420-9. doi: 10.1109/CVPR.2018.00788. URL https://ieeexplore.ieee.org/document/8578886/.

Jianqi Ma, Weiyuan Shao, Hao Ye, Li Wang, Hong Wang, Yingbin Zheng, and Xiangyang Xue. Arbitrary-Oriented Scene Text Detection via Rotation Proposals. *IEEE Transactions on Multimedia*, 20(11):3111–3122, November 2018. ISSN 1941-0077. doi: 10.1109/TMM.2018.2818020. Conference Name: IEEE Transactions on Multimedia.

Anand Mishra, Karteek Alahari, and Cv Jawahar. Scene Text Recognition using Higher Order Language Priors. In *Procedings of the British Machine Vision Conference 2012*, pages 127.1–127.11, Surrey, 2012. British Machine Vision Association. ISBN 978-1-901725-46-9. doi: 10.5244/C.26.127. URL http://www.bmva.org/bmvc/2012/BMVC/paper127/index.html.

Tom M. Mitchell. *Machine Learning*. McGraw-Hill series in computer science. McGraw-Hill, New York, 1997. ISBN 978-0-07-042807-2.

Koji Nakamichi, Kyoko Ohashi, Isao Namba, Rieko Yamamoto, Mikio Aoyama, Lisa Joeckel, Julien Siebert, and Jens Heidrich. Requirements-Driven

Method to Determine Quality Characteristics and Measurements for Machine Learning Software and Its Evaluation. In *2020 IEEE 28th International Requirements Engineering Conference (RE)*, pages 260–270, August 2020. doi: 10.1109/RE48521.2020.00036. ISSN: 2332-6441.

Nibal Nayef, Fei Yin, Imen Bizid, Hyunsoo Choi, Yuan Feng, Dimosthenis Karatzas, Zhenbo Luo, Umapada Pal, Christophe Rigaud, Joseph Chazalon, Wafa Khlif, Muhammad Muzzamil Luqman, Jean-Christophe Burie, Cheng-lin Liu, and Jean-Marc Ogier. ICDAR2017 Robust Reading Challenge on Multi-Lingual Scene Text Detection and Script Identification - RRC-MLT. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 01, pages 1454–1459, November 2017. doi: 10.1109/ICDAR.2017.237. ISSN: 2379-2140.

Wei Niu, Xiaolong Ma, Yanzhi Wang, and Bin Ren. 26ms Inference Time for ResNet-50: Towards Real-Time Execution of all DNNs on Smartphone. *arXiv:1905.00571 [cs, stat]*, May 2019. URL `http://arxiv.org/abs/1905.00571`. arXiv: 1905.00571.

Oyebade K. Oyedotun, Ebenezer O. Olaniyi, and Adnan Khashman. Deep Learning in Character Recognition Considering Pattern Invariance Constraints. *International Journal of Intelligent Systems and Applications*, 7(7): 1–10, June 2015. ISSN 2074904X, 20749058. doi: 10.5815/ijisa.2015.07.01. URL `http://www.mecs-press.org/ijisa/ijisa-v7-n7/v7n7-1.html`.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training Recurrent Neural Networks. *arXiv:1211.5063 [cs]*, February 2013. URL `http://arxiv.org/abs/1211.5063`. arXiv: 1211.5063.

Moacir Antonelli Ponti, Leonardo Sampaio Ferraz Ribeiro, Tiago Santana Nazare, Tu Bui, and John Collomosse. Everything You Wanted to Know about Deep Learning for Computer Vision but Were Afraid to Ask. In *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T)*, pages 17–41, October 2017. doi: 10.1109/SIBGRAPI-T.2017.12. ISSN: 2474-0705.

Liang Qiao, Sanli Tang, Zhanzhan Cheng, Yunlu Xu, Yi Niu, Shiliang Pu, and Fei Wu. Text Perceptron: Towards End-to-End Arbitrary-Shaped Text Spotting. *arXiv:2002.06820 [cs]*, October 2021. URL `http://arxiv.org/abs/2002.06820`. arXiv: 2002.06820.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. pages 779–788, 2016.

URL https://www.cv-foundation.org/openaccess/content_cvpr_2016/html/Redmon_You_Only_Look_CVPR_2016_paper.html.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper/2015/hash/14bfa6bb14875e45bba028a21ed38046-Abstract.html.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *arXiv:1506.01497 [cs]*, January 2016. URL http://arxiv.org/abs/1506.01497. arXiv: 1506.01497.

Tao Sheng, Jie Chen, and Zhouhui Lian. CentripetalText: An Efficient Text Instance Representation for Scene Text Detection. *arXiv:2107.05945 [cs]*, October 2021. URL http://arxiv.org/abs/2107.05945. arXiv: 2107.05945.

Alex Sherstinsky. Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network. *Physica D: Nonlinear Phenomena*, 404:132306, March 2020. ISSN 01672789. doi: 10.1016/j.physd.2019.132306. URL http://arxiv.org/abs/1808.03314. arXiv: 1808.03314.

Baoguang Shi, Xiang Bai, and Serge Belongie. Detecting Oriented Text in Natural Images by Linking Segments. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3482–3490, Honolulu, HI, July 2017a. IEEE. ISBN 978-1-5386-0457-1. doi: 10.1109/CVPR.2017.371. URL http://ieeexplore.ieee.org/document/8099854/.

Baoguang Shi, Xiang Bai, and Cong Yao. An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(11):2298–2304, November 2017b. ISSN 1939-3539. doi: 10.1109/TPAMI.2016.2646371. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

Baoguang Shi, Cong Yao, Minghui Liao, Mingkun Yang, Pei Xu, Linyan Cui, Serge Belongie, Shijian Lu, and Xiang Bai. ICDAR2017 Competition on Reading Chinese Text in the Wild (RCTW-17). In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 01, pages 1429–1434, November 2017c. doi: 10.1109/ICDAR.2017.233. ISSN: 2379-2140.

Baoguang Shi, Mingkun Yang, Xinggang Wang, Pengyuan Lyu, Cong Yao, and Xiang Bai. ASTER: An Attentional Scene Text Recognizer with Flexible Rectification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(9):2035–2048, September 2019. ISSN 1939-3539. doi: 10.1109/TPAMI.2018.2848939. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

Ajay Shrestha and Ausif Mahmood. Review of Deep Learning Algorithms and Architectures. *IEEE Access*, 7:53040–53065, 2019. ISSN 2169-3536. doi: 10.1109/ACCESS.2019.2912200. Conference Name: IEEE Access.

Julien Siebert, Lisa Joeckel, Jens Heidrich, Adam Trendowicz, Koji Nakamichi, Kyoko Ohashi, Isao Namba, Rieko Yamamoto, and Mikio Aoyama. Construction of a quality model for machine learning systems. *Software Quality Journal*, June 2021. ISSN 0963-9314, 1573-1367. doi: 10.1007/s11219-021-09557-y. URL `https://link.springer.com/10.1007/s11219-021-09557-y`.

Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv:1409.1556 [cs]*, April 2015. URL `http://arxiv.org/abs/1409.1556`. arXiv: 1409.1556.

Hannah Snyder. Literature review as a research methodology: An overview and guidelines. *Journal of Business Research*, 104:333–339, November 2019. ISSN 0148-2963. doi: 10.1016/j.jbusres.2019.07.039. URL `http://www.sciencedirect.com/science/article/pii/S0148296319304564`.

Nikolaos Sourvanos and Georgios Tsatiris. Challenges in Input Preprocessing for Mobile OCR Applications: A Realistic Testing Scenario. In *2018 9th International Conference on Information, Intelligence, Systems and Applications (IISA)*, pages 1–5, July 2018. doi: 10.1109/IISA.2018.8633688.

Wanhua Su, Yan Yuan, and Mu Zhu. A Relationship between the Average Precision and the Area Under the ROC Curve. In *Proceedings of the 2015 International Conference on The Theory of Information Retrieval*, pages 349–352, Northampton Massachusetts USA, September 2015. ACM. ISBN 978-1-4503-3833-2. doi: 10.1145/2808194.2809481. URL `https://dl.acm.org/doi/10.1145/2808194.2809481`.

Yipeng Sun, Zihan Ni, Chee-Kheng Chng, Yuliang Liu, Canjie Luo, Chun Chet Ng, Junyu Han, Errui Ding, Jingtuo Liu, Dimosthenis Karatzas, Chee Seng Chan, and Lianwen Jin. ICDAR 2019 Competition on Large-Scale Street

View Text with Partial Labeling - RRC-LSVT. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1557–1562, September 2019. doi: 10.1109/ICDAR.2019.00250. ISSN: 2379-2140.

Richard J. Torraco. Writing Integrative Literature Reviews: Guidelines and Examples. *Human Resource Development Review*, 4(3):356–367, September 2005. ISSN 1534-4843. doi: 10.1177/1534484305278283. URL `https://doi.org/10.1177/1534484305278283`. Publisher: SAGE Publications.

Andreas Vogelsang and Markus Borg. Requirements Engineering for Machine Learning: Perspectives from Data Scientists. In *2019 IEEE 27th International Requirements Engineering Conference Workshops (REW)*, pages 245–251, September 2019. doi: 10.1109/REW.2019.00050.

Zhaoyi Wan, Minghang He, Haoran Chen, Xiang Bai, and Cong Yao. TextScanner: Reading Characters in Order for Robust Scene Text Recognition. *arXiv:1912.12422 [cs]*, January 2020a. URL `http://arxiv.org/abs/1912.12422`. arXiv: 1912.12422.

Zhaoyi Wan, Jielei Zhang, Liang Zhang, Jiebo Luo, and Cong Yao. On Vocabulary Reliance in Scene Text Recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11422–11431, Seattle, WA, USA, June 2020b. IEEE. ISBN 978-1-72817-168-5. doi: 10.1109/CVPR42600.2020.01144. URL `https://ieeexplore.ieee.org/document/9156361/`.

Wenhai Wang, Enze Xie, Xiang Li, Wenbo Hou, Tong Lu, Gang Yu, and Shuai Shao. Shape Robust Text Detection With Progressive Scale Expansion Network. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9328–9337, Long Beach, CA, USA, June 2019a. IEEE. ISBN 978-1-72813-293-8. doi: 10.1109/CVPR.2019.00956. URL `https://ieeexplore.ieee.org/document/8953886/`.

Wenhai Wang, Enze Xie, Xiaoge Song, Yuhang Zang, Wenjia Wang, Tong Lu, Gang Yu, and Chunhua Shen. Efficient and Accurate Arbitrary-Shaped Text Detection With Pixel Aggregation Network. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8439–8448, Seoul, Korea (South), October 2019b. IEEE. ISBN 978-1-72814-803-8. doi: 10.1109/ICCV.2019.00853. URL `https://ieeexplore.ieee.org/document/9009483/`.

Enze Xie, Yuhang Zang, Shuai Shao, Gang Yu, Cong Yao, and Guangyao Li. Scene Text Detection with Supervised Pyramid Context Network.

arXiv:1811.08605 [cs], November 2018. URL `http://arxiv.org/abs/1811.08605`. arXiv: 1811.08605.

Zecheng Xie, Yaoxiong Huang, Yuanzhi Zhu, Lianwen Jin, Yuliang Liu, and Lele Xie. Aggregation Cross-Entropy for Sequence Recognition. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6531–6540, Long Beach, CA, USA, June 2019. IEEE. ISBN 978-1-72813-293-8. doi: 10.1109/CVPR.2019.00670. URL `https://ieeexplore.ieee.org/document/8953200/`.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. *arXiv:1502.03044 [cs]*, April 2016. URL `http://arxiv.org/abs/1502.03044`. arXiv: 1502.03044.

Yongchao Xu, Yukang Wang, Wei Zhou, Yongpan Wang, Zhibo Yang, and Xiang Bai. TextField: Learning a Deep Direction Field for Irregular Scene Text Detection. *IEEE Transactions on Image Processing*, 28(11):5566–5579, November 2019. ISSN 1941-0042. doi: 10.1109/TIP.2019.2900589. Conference Name: IEEE Transactions on Image Processing.

Chuhui Xue, Shijian Lu, and Fangneng Zhan. Accurate Scene Text Detection through Border Semantics Awareness and Bootstrapping. *arXiv:1807.03547 [cs]*, July 2018. URL `http://arxiv.org/abs/1807.03547`. arXiv: 1807.03547 version: 3.

Xue Yang, Xiaojiang Yang, Jirui Yang, Qi Ming, Wentao Wang, Qi Tian, and Junchi Yan. Learning High-Precision Bounding Box for Rotated Object Detection via Kullback-Leibler Divergence. *arXiv:2106.01883 [cs]*, October 2021. URL `http://arxiv.org/abs/2106.01883`. arXiv: 2106.01883.

Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures. *Neural Computation*, 31(7):1235–1270, July 2019. ISSN 0899-7667, 1530-888X. doi: 10.1162/neco_a_01199. URL `https://direct.mit.edu/neco/article/31/7/1235-1270/8500`.

Liu Yuliang, Jin Lianwen, Zhang Shuaitao, and Zhang Sheng. Detecting Curve Text in the Wild: New Dataset and New Solution. *arXiv:1712.02170 [cs]*, December 2017. URL `http://arxiv.org/abs/1712.02170`. arXiv: 1712.02170 version: 1.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. Recurrent Neural Network Regularization. *arXiv:1409.2329 [cs]*, February 2015. URL `http://arxiv.org/abs/1409.2329`. arXiv: 1409.2329.

Fangneng Zhan and Shijian Lu. ESIR: End-To-End Scene Text Recognition via Iterative Image Rectification. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2054–2063, Long Beach, CA, USA, June 2019. IEEE. ISBN 978-1-72813-293-8. doi: 10.1109/CVPR.2019.00216. URL `https://ieeexplore.ieee.org/document/8954060/`.

Jie M. Zhang, Mark Harman, Lei Ma, and Yang Liu. Machine Learning Testing: Survey, Landscapes and Horizons. *IEEE Transactions on Software Engineering*, pages 1–1, 2020. ISSN 1939-3520. doi: 10.1109/TSE.2019.2962027. Conference Name: IEEE Transactions on Software Engineering.

Zhenyao Zhao, Min Jiang, Shihui Guo, Zhenzhong Wang, Fei Chao, and Kay Chen Tan. Improving Deep Learning based Optical Character Recognition via Neural Architecture Search. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–7, July 2020. doi: 10.1109/CEC48606.2020.9185798.

Didar Zowghi, Zhi Jin, Simone Diniz Junqueira Barbosa, Phoebe Chen, Alfredo Cuzzocrea, Xiaoyong Du, Joaquim Filipe, Orhun Kara, Igor Kotenko, Krishna M. Sivalingam, Dominik Ślęzak, Takashi Washio, and Xiaokang Yang, editors. *Requirements Engineering*, volume 432 of *Communications in Computer and Information Science*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014. ISBN 978-3-662-43609-7 978-3-662-43610-3. doi: 10.1007/978-3-662-43610-3. URL `http://link.springer.com/10.1007/978-3-662-43610-3`.

Appendix A

Benchmark Dataset Examples

Figure A.1 shows representational examples taken out of benchmark datasets for STD, STR and STS.

(a) ICDAR (2013) (Karatzas et al., 2013)

(b) ICDAR (2015) (Karatzas et al., 2015)

(c) ICDAR MLT (2017) (Nayef et al., 2017)

(d) IIIT 5k Word (2012) (Mishra et al., 2012)

(e) MSRA-TD500 (Cong Yao et al., 2012)

(f) Total Text (2017) (Ch'ng and Chan, 2017)

(g) SCUT CTW1500 (2017) (Yuliang et al., 2017)

Figure A.1: Benchmark data set examples

Appendix B

Litaratur Qualities

The following tables show qualities that where identified in literature. The qualities are categorized by the MLS entities defined in Siebert et al. (2021). The model entity is the focus of this work and is thus discussed in Chapter 4.

Qualitiy	**Sources**
Relevancy	Ashmore et al. (2021)
Currentness	Siebert et al. (2021)
Completeness	Ashmore et al. (2021); Vogelsang and Borg (2019); Siebert et al. (2021)
Balancedness	Ashmore et al. (2021); Siebert et al. (2021)
Consistency	Vogelsang and Borg (2019)
Intra-Consistency	Siebert et al. (2021)
Inter-Consistency	Siebert et al. (2021)
Accuracy	Ashmore et al. (2021)
Absence of bias	Siebert et al. (2021)
Correctness	Vogelsang and Borg (2019)
Data Representativeness	Nakamichi et al. (2020); Siebert et al. (2021)
Suitability of Training Data	Nakamichi et al. (2020)
Test Dataset Creating Appropriateness	Nakamichi et al. (2020)
Independence of Train and Test Data	Nakamichi et al. (2020); Siebert et al. (2021)

Table B.1: MLS qualities identified for data entity

Qualitiy	**Sources**
Capacity of Data Storage	Nakamichi et al. (2020)
Infrastructure suitability	Siebert et al. (2021)
Deployment Fit-for-Purpose	Ashmore et al. (2021)
Training Process Appropriateness	Nakamichi et al. (2020)
Training efficiency	Siebert et al. (2021)

Table B.2: MLS qualities identified for infrastrucure entity

Qualitiy	Sources
Coverage of Usage Environment	Nakamichi et al. (2020)
Coverage of Operation Environment	Nakamichi et al. (2020)
Scope compliance	Siebert et al. (2021)
Social impact	Siebert et al. (2021)
Environmental Impact of training process	Siebert et al. (2021)
Contextual Relevancy	Ashmore et al. (2021)

Table B.3: MLS qualities identified for environment entity

Qualitiy	Sources
Suitability of Input Data Quality Maintenance	Nakamichi et al. (2020)
Quality Maintenance for Test Data Appropriateness	Nakamichi et al. (2020)
Security and Privacy Assurance	Nakamichi et al. (2020); Zhang et al. (2020)
Troubleshooting	Arpteg et al. (2018)
Easiness of Resource Update	Nakamichi et al. (2020)
Easiness of Software Update	Nakamichi et al. (2020)
Easiness of System Status Analysis	Nakamichi et al. (2020)
Runtime correctness	Siebert et al. (2021)
Legal and Regularity Requirements	Vogelsang and Borg (2019)
Effectiveness of output supervision	Siebert et al. (2021)
Efficiency of output supervision	Siebert et al. (2021)
Appropriateness of Operation Maintenance	Nakamichi et al. (2020)
Deployment Tolerability	Ashmore et al. (2021)
Deployment Adaptability	Ashmore et al. (2021)

Table B.4: MLS qualities identified for system entity