University of St. Gallen – School of Management,
Economics, Law, Social Sciences, and International Relations
(HSG)

Master's Thesis

ELEA

Building of an adaptive learning support system to foster empathy skills amongst students

Corinne Ruckstuhl
Master of Arts in Business Innovation
[redacted]

Supervisor: Prof. Dr. Matthias Söllner
Co-Supervisor: Prof. Dr. Jan Marco Leimeister
Institute of Information Management (IWI-HSG)
Müller-Friedbergstrasse 8
9000 St. Gallen

St. Gallen, October 2020

“Leadership is about empathy.”

Oprah Winfrey

ABSTRACT

Rapid advancements in technology and globalization drive the need for empathic leadership. Yet, empathy experiences a major decrease amongst students, and educational institutions are not prepared to teach empathy skills due to limited resources in large-scale lecture formats. However, recent advancements in Natural Language Processing offer new possibilities to detect empathy in textual data and foster empathy skills amongst students. In this master's thesis, an Empathy Learning Application (ELEA) was built that provides students with adaptive and individual feedback on their empathy level. Empathy is the ability to react to the observed experiences of another and to simply understand the other person's perspective. The responsive writing-support system was built in three design cycles, following the Design Science Research approach. First, a corpus to model empathy in student-written peer reviews was created, second, two artificial neural networks to predict empathy based on bidirectional encoder representations from transformers were developed, and third, a responsive writing-support system to provide students with adaptive feedback on their individual empathy learning journey was designed. User experiments reveal significant higher empathy skill learning with ELEA compared to an alternative tool. Additionally, ELEA's high technology acceptance amongst students indicates promising results to incorporate the empathy learning application to further educational settings and foster empathy skills amongst students.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
ANN	Artificial Neural Network
BERT	Bidirectional Encoder Representations from Transformers
CML	Chatbot-mediated Learning
CPM	Confusion Probability Matrix
DL	Deep Learning
DSR	Design Science Research
FARM	Framework for Adapting Representation Models
Colab	Google Colaboratory
GRU	Gated Recurrent Unit
IAA	Inter Annotator Agreement
ITPA	IT-based Peer Assessment
LSTM	Long Short-Term Memory
MAMA	Model, Annotate, Model, Annotate
MATTER	Model, Annotate, Train, Test, Evaluate, Revise
MITI	Motivational Interviewing Treatment Integrity
ML	Machine Learning
MOOC	Massive Open Online Course
NLP	Natural Language Processing
RL	Reinforcement Learning
SL	Supervised Learning
SSL	Semi-supervised Learning
TMLS	Technology-mediated Learning Services
UNESCO	United Nations Educational, Scientific and Cultural Organization
UL	Unsupervised Learning

1 INTRODUCTION

The rapid progress of today's technology and globalization leads to many new challenges. Societies around the world struggle to keep pace with the rising complexity and uncertainty, educational organizations are confronted with new learning requirements, and individuals are exposed to an increasing amount of information on a daily basis. It is no surprise, that the United Nations Educational, Scientific and Cultural Organization (UNESCO) published the Global Education Agenda 2030 which looks towards a more sustainable, inclusive and diverse education, in which individuals are taught to be 'change-makers' (UNESCO, 2017, p. 7). To become a change-maker and leader for tomorrow, certain skills, knowledge, values and attitudes are required to empower others and face challenges. The Global Education Agenda 2030 expounds collaborative competencies such as *empathy* and *empathic leadership* as one of the key competencies towards a more sustainable future (UNESCO, 2017, p. 10). Empathy and empathic leadership are about understanding the perspectives and actions of others, being sensitive to others, dealing with conflicts in groups, and facilitating collaborative and participatory problem solving. All this is required to be successful in today's fast-moving world (UNESCO, 2017).

Notwithstanding the importance of empathy for future leaders, empathy skills amongst students have decreased rapidly in the period between 2000–2009 (e. g. Konrath, O'Brien, & Hsing, 2011) and are expected to drop even more for the period between 2009–2019 (Kaitlin & Konrath, 2019). It is therefore crucial to further incorporate empathy skills to the educational system and promote empathy skills amongst students. However, teaching empathy requires many resources from educators such as continuous support and individual feedback (Hattie & Timperley, 2007). Especially in higher education systems where new learning formats such as Massive Open Online Courses (MOOCs) and large-scale lectures are in place, this can hardly be provided.

However, advancements in new technologies such as Artificial Intelligence (AI) and Natural Language Processing (NLP) provide solutions for these problems. Such intelligent support systems are able to provide individual feedback to students, assisting professors in promoting and educating specific skills.

1.1 Research Question and Research Objective

One possibility to leverage technology to promote empathy skills amongst students is the use of empathy detection on natural language texts to identify and model empathic structures. However, this research field emerged just recently and has made little contribution, as a body of research or in educational practice, yet. Only few studies focus on the prediction of empathy in natural language texts (e. g. Xiao, Imel, Georgiou, Atkins, & Narayanan, 2015; Khanpour, Caragea, & Biyani, 2017; Buechel, Buffone, Slaff, Ungar, & Sedoc, 2018). This might be due to the complexity of the construct of empathy, its various psychological perspectives and its interdependence with body gestures or language characteristics (Buechel, Buffone, Slaff, Ungar, & Sedoc, 2018). Moreover, to the best of the author's knowledge, no adaptive writing-support system has been developed yet, that supports students in improving their

empathy skills based on *textual* data in a pedagogical scenario.

This master's thesis therefore aims to develop a writing-support system to automatically detect empathy in natural language texts and provide adaptive learning feedback to students. Particularly, this research project focus on a pedagogical scenario which is based on German-written business models and peer reviews. By applying the Design Science Research (DSR) approach proposed by Hevner (2007), this master's thesis contributes to research by answering the following three research questions:

#	Research Question
1	How can a corpus for modeling empathy in German student-written peer reviews be developed?
2	How can artificial intelligence be used to detect and predict empathy in German student-written peer reviews?
3	How can an effective and user-centered writing-support system be created to support students in improving their empathy skills in peer reviews?

Table 1: Research questions

1.2 Thesis Structure

In order to answer the above research questions, the thesis is structured in five parts (see Figure 1). The first part outlines crucial theoretical and conceptual background information that is needed to understand this thesis. This includes knowledge about empathy, feedback, Technology-mediated Learning Services (TMLS), and details on text mining and NLP. The second part covers relevant related work, both in emotion recognition and empathy detection. Thirdly, the methodology used in this thesis is presented and explained. The fourth part documents the three design cycles of this research project. This includes 1) the corpus development, 2) the modeling of deep neural networks to predict empathy, and 3) the creation of ELEA, a user-centered writing-support system to support students in improving their empathy in peer reviews. The thesis ends with a conclusion, limitations and future research topics.

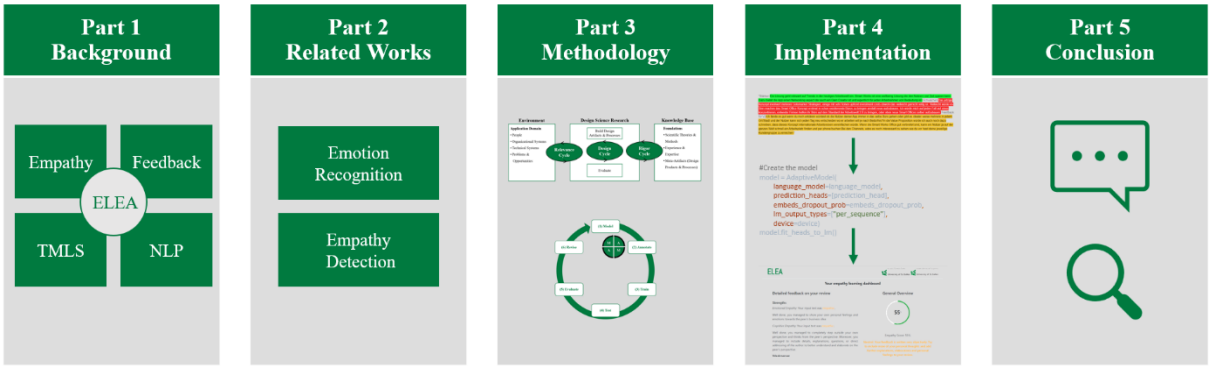


Figure 1: Thesis structure (own illustration)

2 THEORETICAL AND CONCEPTUAL BACKGROUND

The aim of the following chapter is to provide a basic understanding of the theoretical foundations of this thesis. First, this includes a detailed explanation and description of the concept of *empathy*. Additionally, with the pedagogical scenario including peer reviews from German business models, a short overview about the construct *feedback* is provided, too. Thirdly, the reader gains insights into *technology-mediated learning services*, such as writing-support systems. Finally, this chapter also includes import conceptual details about the technologies and techniques being used for the development of the final prototype ELEA. This particularly includes information on *NLP, Machine Learning (ML)* and *Artificial Neural Networks (ANN).*

2.1 Empathy

Ever since the term *empathy* has been associated with Titchener's German word *Einfühlung* (Titchener, 1909; Wispé, 1987), the construct of empathy has been considered as an important component of social cognition that contributes to the human ability of understanding and responding adaptively to other's emotions (Spreng, McKinnon, Mar, & Levine, 2009). Originally translated and understood as "feeling into", empathy nowadays counts as many definitions as there are authors in the field (De Vignemont & Singer, 2006, p. 1; Decety & Jackson, 2004, p. 1). Researchers claim that there is a lack of a clear, universal definition for empathy (Neumann, Chan, Boyle, Wang, & Westbury, 2015, p. 1). The following table shows frequently used definitions of empathy from research in chronological order.

Author(s)	**Definition**
Hoffman (1982)	"An affective response more appropriate to someone else's situation than to one's own."
Davis (1983)	"Reactions of one individual to the observed experiences of another […] and simply understanding the other person's perspective. "
Goldman (1993)	"The ability to put oneself into the mental shoes of another person to understand his or her emotions and feelings."
Ickes (1997)	"A complex form of psychological inference in which observation, memory, knowledge, and reasoning are combined to yield insights into the thoughts and feelings of others."
Batson et al. (1997)	"An other-oriented emotional response congruent with the other's perceived welfare."
Eisenberg (2000)	"An affective response that stems from the apprehension or comprehension of another's emotional state or condition, and which is similar to what the other person is feeling or would be expected to feel in the given situation."
Baron-Cohen & Wheelwright (2004)	"The drive or ability to attribute mental states to another person/animal, and entails an appropriate affective response in the observer to the other

	person's mental state."
Decety & Jackson (2004)	"The ability to experience and understand what others feel without confusion between oneself and others."
Spreng et al. (2009)	"An emotional process, or an accurate affective insight into the feeling state of another."
Oliveira-Silva & Gonçalves (2011)	"The capacities to resonate with another person's emotions, understand his/her thoughts and feelings, separate our own thoughts and emotions from those of the observed and responding with the appropriate prosocial and helpful behaviour."

Table 2: Different definitions of empathy

Besides defining what empathy is, there has been extensive work on how to measure it. Many researchers focus on questionnaires (mostly self-report measures) to measure empathy, but other possibilities include neuroscientific or behavioral measures. Neuroscientific measures cover Magnetic Resonance Imaging (e. g. Bergemann, 2009) or Facial Electromyography (e. g. Westbury & Neumann, 2008) amongst others, and behavioral measures include the Kids Empathetic Development Scale (Reid et al., 2012) for example. The following tables summarizes the most used empathy questionnaires in chronological order (list is not exhaustive).

Name and Author(s)	**Summary**
Hogan Empathy Scale (Hogan, 1969)	64-item scale composed of 31 items selected from the Minnesota Multiphasic Personality Inventory (Hathaway & McKinley, 1943), 25 items selected from the California Psychological Inventory (Gough, 1964) and 8 items created by Hogan and colleagues. It contains four separate dimensions: social self-confidence, even-temperedness, sensitivity, and nonconformity.
Questionnaire Measure of Emotional Empathy (Mehrabian & Epstein, 1972)	33 items using 9-point ratings from -4 = very strong disagreement to +4 = very strong agreement. Contains seven subscales: Susceptibility to emotional contagion, appreciation of the feelings of unfamiliar and distant others, extreme emotional responsiveness, tendency to be moved by others' positive emotional experiences, tendency to be moved by others' negative emotional experiences, sympathetic tendency, and willingness to be in contact with others who have problems.
Interpersonal Reactivity Index (Davis, 1983)	28-items answered on a 5-point Likert scale ranging from "Does not describe me well" to "Describes me very well". Contains four subscales, each made up of 7 different items: Perspective Taking and Fantasy in addition to Empathic Concern and Personal Distress
Balanced Emotional Empathy Scale (Mehrabian, 1996)	30 items rated on a 9-point Likert-type scale ranging from +4 = very strong agreement to -4 = very strong disagreement. The scale yields a single score with higher scores representing greater levels of emotional empathy.
Empathy Quotient	60-item scale with 40 empathy items and 20 control items. Scale ranges from "strongly agree" to "disagree strongly". Control items are used to

(Baron-Cohen & Wheelwright, 2004)	provide some distraction and check response bias. Both affective and cognitive empathy is included in the survey.
Basic Empathy Scale (Joliffe & Farrington, 2006)	40-item scale with reverse worded items being included. 20 items require a positive response and 20 items require a negative response. Items measure five basic emotions (fear, sadness, anger, joy and happiness) wherein the measurements relate more generally to cognitive and affective empathy and not to a non-specific affective state (e.g., anxiety).
Toronto Empathy Questionnaire (Spreng et al., 2009)	Consists of 16 items, each rated on a five point scale from 'never' to 'always' with an equal number of positively and reverse worded items. It was developed by reviewing other available empathy instruments and determining what these instruments had in common. The TEQ loads on a single factor representative of 'the broadest, common construct of empathy'.
Questionnaire of Cognitive and Affective Empathy (Reniers, Corcoran, Drake, Völlm, & Shryane, 2011)	31-item measure with a 4-point forced-choice response scale, consisting of five subscales: perspective taking, online simulation, emotion contagion, proximal responsivity, and peripheral responsivity. The first two subscales measure cognitive empathy while the remaining three subscales measure affective empathy. The items were assessed and derived from previous questionnaires.

Table 3: Different measurements of empathy

Pondering these definitions and measurements, it becomes clear that the construct of empathy can be viewed and understood from different perspectives. Cuff, Taylor, Brown, & Howat (2016) have identified eight "areas of confusion" (p. 4) that play an important role in understanding the construct of empathy. These areas are: 1) distinguishing empathy from other concepts, 2) cognitive or affective?, 3) congruent or incongruent?, 4) subject to other stimuli?, 5) self-other distinction or merging?, 6) trait or state influences?, 7) has a behavioral outcome?, and 8) automatic or controlled?

For the sake of this thesis and the further development of this work, it is sufficient to clear the air about 1) and 2).

Distinguishing empathy from other concepts

Perhaps one of the most vivid discussions in the field of emotional research is the relationship between empathy and concepts such as sympathy, compassion or tenderness (see e. g. Batson, 2011; Eisenberg, Shea, Carlo, & Knight, 1991; Preston & de Waal, 2002). Several authors appear to merge the different concepts (see e. g. Barnett & Mann, 2013; Pavey, Greitemeyer, & Sparks, 2012; Preston & De Waal, 2002; Singer & Lamm, 2009). However, many others claim functional differences between empathy and related concepts (see e. g. Eisenberg et al., 1991; Hein & Singer 2008). This functional difference is defined as "feeling *as* and feeling *for* the other" (Singer & Lamm, 2009, p. 157), whereas empathy is provoking the same emotion and sympathy a different emotion in the observer. Furthermore, compassion ("the feeling that arises in witnessing another's suffering and that motivates a subsequent desire to help" (Goetz, Keltner, & Simon-Thomas, 2010, p. 351)) and tenderness ("an expansive, "warm-and-fuzzy" feeling often elicited by the delicate and defenceless" (Lishner, Batson, & Huss, 2011, p. 615)) can both be distinguished from empathy due to their concentration on feelings *towards* another person

rather than the sharing of emotions *with* a person (Cuff et al., 2016, pp. 6–7). Although a distinction is not always clear, this thesis acknowledges the functional differences between the concepts and therefore follows the second research stream.

Cognitive or affective?

Table 2 clearly demonstrates that empathy can be divided into various components and subscales, where some of them include both cognitive and emotional (affective) components, but others are based upon only cognitive or emotional parts. As early as 1980, Davis referred to an integration of the two research traditions and claimed that the two components "compromise an interdependent system in which each influences the other" (Davis, 1980, p. 3). Today's widely accepted understanding of empathy also includes both emotional (affective) and cognitive empathy (Decety & Jackson, 2004; Lawrence, Shaw, Baker, Baron-Cohen & David, 2004; Jolliffe & Farrington, 2006; Gini, Albiero, Benelli & Altoe, 2007; Spreng et al., 2009). Being aware that there are multiple perspectives on empathy, this thesis includes both the cognitive and emotional component of empathy (according to Lawrence et al., 2004) and therefore follows the *Interpersonal Reactivity Index* (Davis, 1983), *Empathy Quotient* (Baron-Cohen & Wheelwright, 2004), *Toronto Empathy Questionnaire* (Spreng et al., 2009), and the *Questionnaire of Cognitive and Affective Empathy* (Reniers et al., 2011). Cognitive empathy has been related to Davis' Perspective Taking (1980, p. 6) and means the ability to use cognitive processes such as role taking, perspective taking or decentering. A person sets aside their own perspective and steps into the shoes of the other. Cognitive empathy can be purely cognitive in that there is no reference to any affective state, but mostly includes understanding the other's emotional state as well (Baron-Cohen & Wheelwright, 2004, p. 164). Emotional empathy has been related to Davis' Empathic Concern (1980, p. 6) and concerns the experience of emotion. An experience of emotion or an emotional response towards another person can either be a feeling that matches exactly the feelings of the observed person, simply an appropriate feeling to the other person's emotional state, or that the feeling in the observer must be one of concern or compassion to another's persons distress (Baron-Cohen & Wheelwright, 2004, p. 164).

In conclusion, there is no strict consensus in defining and measuring empathy. However, since the aim of this thesis is not to reconcile the differences between the several definitions and find a consensus, the above-mentioned elaborations about empathy are exhaustive enough.In this thesis, empathy is defined as the "***ability to react to the observed experiences of another […] and simply understand the other person's perspective*** "(Davis 1983, p. 1), where empathy consists of both ***emotional*** and ***cognitive*** components (Spreng et al., 2009).

2.2 Feedback

Feedback is defined as "information provided by an agent […] regarding aspects of one's performance or understanding" (Hattie & Timperley, 2007, p. 102). Feedback can either be generated from agents such as teachers, parents, or peers, or internally self-generated (Butler & Winnie, 1995). The aim of feedback is to close the gap between the current and the pursued understanding (Sadler, 1989). Ramaprasad (1983) explains that "information on the gap when used to alter the gap (most probably to decrease the gap) becomes feedback" (p. 5). Affective and cognitive processes can help to decrease this gap. Affective processes include raised effort, ambition, or engagement, whereas cognitive processes consist of validating results, providing additional information, giving directions, or showing different ways to reach the aimed understanding. Typically, feedback consists of three parts: 1) elaboration of strengths, 2) elaboration of weaknesses and 3) suggestions for improvements (Hattie & Timperley, 2007). However, constructive feedback not only contains these components, but elaborates on them with explanations. Therefore, argumentation theory can be added to the established model of feedback components (see Toulmin, Rieke, & Janik (1984) for an introduction to reasoning). According to a minimal definition, an argument is a set of statements made up of claims and premises (Walton, 2009, p. 2). The claim represents the conclusion of an argument and exemplifies the opinion or a point of view that someone has claimed. Sentences that support the validity of the claim are called the premise (Johnson & Blair, 1994, p. 10). In that sense, the claim represents the central component of the feedback that is backed up with elaborations, examples, or explanations as the premise.

Many research studies have shown that providing feedback has positive effects on the learning results and performance of feedback receivers. Pavett (1983) has proven that communication in form of feedback has a significant positive influence on the performance of employees (p. 650). Ten years later, Karl, O'Leary-Kelly, and Martocchio (1993) confirmed these findings in their experiment in a speed-reading class. They concluded that students who received feedback on their performance experienced significantly greater increases in self-efficacy than students in subjects who did not received any feedback (Karl, O'Leary-Kelly, & Martocchio, 1993, p. 379). Furthermore, Vollmeyer and Rheinberg (2005) found out, that knowledge acquisition and application increased with feedback provided during solving exercises (p. 599). Liu & Carless (2006) claim several arguments on how students can learn not only from peer review as a feedback technique itself, but through meta-cognitive processes provoked by peer reviews (such as critical reflecting or listening) (p. 289). Lehmann, Söllner & Leimeister (2015) conducted an experiment with IT-based Peer Assessments (ITPA) with students and proved that peer reviews support the student's learning process positively by giving them the opportunity to compare their approach or receive alternative viewpoints on their solution. Generally, ITPAs have been designed to overcome the challenge of providing regular feedback in large group teaching scenarios (e.g. lectures). ITPAs aim to enable students in assessing their current state of knowledge on the one hand, but on the other hand, it helps students to improve their ability to provide feedback (Lehmann, Söllner, & Leimeister, 2015, pp. 1–2). ITPAs consist of several process steps that include self-assessment, the creation of

the assignment, but also a peer review of the assignment. In case of the IT-based review, the IT-System distributes the assignments to a defined amount of anonymous peer students for reviewing (3 in case of Rietsche, Lehmann, Haas, & Söllner, 2017). Each student carries out a peer assessment of the received assignment. After submitting the reviews, each student receives back the evaluated assignment and gets the chance to revise it and submit a second version of it. Figure 2 shows the ITPA process on a high level. The data domain of this thesis is based on IT-based peer reviews (more details in chapter 5).

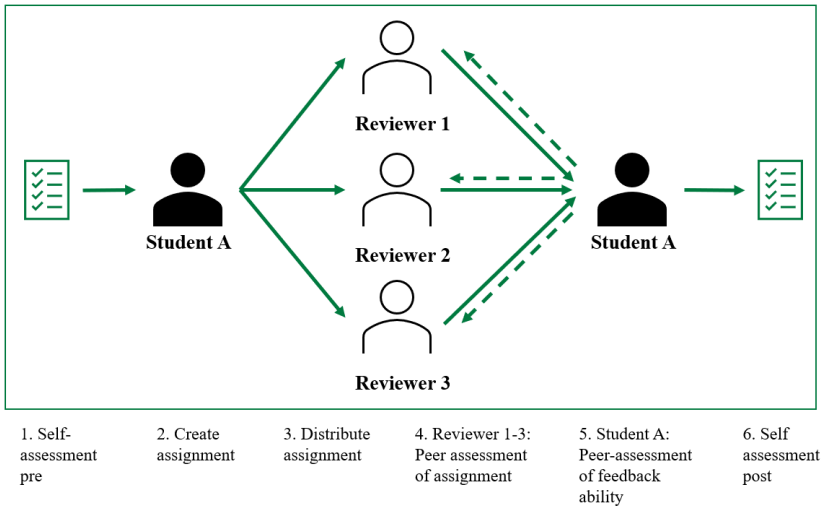


Figure 2: Illustration of the ITPA process (own illustration based on Rietsche et al., 2017)

2.3 Technology-Mediated Learning Services

Rapid technological advancements such as AI, computational linguistic, or conversational agents have brought new chances to both teaching and learning (Webster & Hackley, 1997, p. 1282). For decades, educational technology was a central part of research in disciplines such as psychology, education, business, or information systems leading to the emergence of technology-mediated learning services (Gupta, Bostrom, & Huber, 2010). TMLS can be defined as *"an environment in which the learner's interactions with learning materials (readings, assignments, exercises, etc.), peers, and/or instructors are mediated through advanced information technologies"* (Alavi & Leidner, 2001, p. 2). The above mentioned ITPAs can be classified as such an environment. TMLS describes the concept of how technology influences current learning scenarios and is considered to be a key success factor for innovative learning (Janson & Thiel de Gafenco, 2015, p. 1). Experiments revealed other beneficial aspects of TMLS such as consistency, scalability, cost reduction, better availability, or improved learning outcomes (Lopez-Perez, Perez-Lopez & Rodriguez-Ariza, 2011, p. 818). Moreover, state-of-the-art TMLSs include the possibility to follow and monitor every student's learning state and adapt the learning experience to each individual's needs (Roschelle, 2013, p. 67).

In research, the term TMLS has often been replaced by the term *e-learning* (Gupta & Bostrom, 2013, p. 454). However, in practice, TMLS works in many forms and combines different learning modes and methods. Gupta & Bostrom (2009) identified the following elements as design approaches for such

learning scenarios:

- Web- or computer-based approaches
- Asynchronous or synchronous approaches
- Instructor-led or self-paced approaches
- Individual-based or team-based learning modes

Combining these elements, different TMLS have been developed and constructed. Amongst others, v*irtual reality learning tools* can support students to experience real life scenarios in a secure setting (e.g. practicing medical operations) (Bailenson et al., 2008), *smart personal assistants* are used to support multi-user interaction in complex problem settings (Winkler, Büchi & Söllner, 2019) or, *chatbot-mediated learning* (CML) services enable humans to make a conversation with a computer via text or voice interaction (Toxtli, Monroy-Hernanandez & Cranshaw, 2018). Similar to CML, *writing-support systems* react to a given input of a user. However, the interaction is unidirectional: a user inputs textual data, whereas the writing-support system reacts with an appropriate analysis and adaptive feedback to the input. There is no conversation established yet. Nevertheless, writing-support systems were proven to be very effective in supporting humans to improve their writing (Makarenkov, Rokach, & Shapira, 2019).

2.4 Text Mining and Natural Language Processing

As a subcategory of data mining, text mining provides a way to make use of textual data by combining state-of-the-art techniques such as Machine Learning, Deep Learning (DL), or Natural Language Processing (Aggarwal, 2015, p. 1). This combination allows text mining to be used in a variety of different applications and use cases, such as information retrieval, clustering, prediction, or evaluation of textual data (Weiss, Indurkhya, Zhang & Damerau, 2005, pp. 7–12). Like data mining, text mining undergoes various steps before being able to further process the data. Important initial steps include data preprocessing and cleaning which can be achieved with the help of natural language processing. NLP has gained significant attention in business applications since a huge amount of data generated is represented in text format (such as customer reviews, email, etc.). NLP therefore aims to analyze naturally written text based on linguistic analysis to achieve the same level of language processing as humans (Liddy, 2001, p. 2). Withing NLP, different levels to access the data can be defined (Liddy, 2001, pp.6–7): 1) morphological; which analyzes a word based on the smallest unit of meaning (also called morphemes), 2) lexical; which interprets the meaning of each individual word by assigning a part-of-speech tag, 3) syntactic; which interprets a structure of a sentence by showing dependencies of words within the sentence, and 4) semantic, which focuses on the interactions among word-meanings to analyze the meaning of a sentence.

One important aspect of text mining and NLP is the transformation of unstructured text data to usable structured numerical features for the algorithm. This is done with feature extraction or word embeddings, which can be described as the vector representation of a certain word. While feature extraction in ML requires a lot of hand-crafted effort to define specific features of a corpus, word embeddings in DL allows faster and efficient word processing since its working with pre-trained language models. A word embedding represents each individual word as a vector in a vector space. The information stored in this vector does not only contain syntactic information, but also such of semantical or relational aspects (Vasilev, Slater, Spacagna, Roelants, & Zocca, 2019, pp. 216–220). Up until now, there exist a vast number of methods to transform word to vectors and there have already been many pre-trained language models developed. Word2Vec (Mikolov, Corrado, Chen, & Dean, 2013), GloVe (Pennington, Socher, & Manning, 2014), and FastText (Bojanowski, Grave, Joulin, & Mikolov, 2017) are amongst the most used pre-trained word embedding methods. The Word2Vec methods includes two different architectures, namely a Skip-gram model and a Bag-of-Words model. Both models use the surrounding words and therefore the context to learn about the word's semantic embedding. The GloVe method (Global Vectors for Word representation) considers a word context on a one to one basis by creating a word-word co-occurrence matrix that includes the probability of the occurrence of a certain word close to another certain word. Lastly, the FastText method is similar to the Word2Vec methods, but words are modeled as a character n-gram. Meaning that for example a 5-letter word is a 5-character n-gram and is modeled as subwords, leading to the possibility of representing entire words as the sum of subvectors (Bojanowski, Grave, Joulin, & Mikolov, 2017). Besides these word embedding methods, one particular technique for NLP has taken a lot of attention during the last two years since its publication. BERT (Bidirectional Encoder Representations from Transformers) (Devlin, Chang, Lee, & Toutanova, 2019) is a language model published by Google. It applies bidirectional training to the transformer (see chapter 2.4.2 for more details) method to language modeling and with this new approach, manages to gain much deeper sense to words and a language. As opposed to other models, BERT reads the input text not only from one direction but from both. Furthermore, the authors used a new technique called masked language modeling as well as next sentence prediction to pretrain the model. The masked language modeling hides a certain word in a sentence and then tries to predict this hidden word, while the aim of the next sentence prediction is to determine if two sentences have a logical connection towards each other. As of now, BERT is considered to be one of the most efficient and effective pre-trained language models and has outperformed many other approaches (Devlin, Chang, Lee, & Toutanova, 2019).

In the following, various approaches of ML and DL that are used within the area of text mining and NLP are described and explained. Specific techniques or methods are explained to the extent necessary to understand this thesis.

2.4.1 Machine Learning

Machine Learning evolved from computational learning theory in AI and is thus counting as a subfield of AI. It can be described as a "collection of algorithms and techniques used to create computational systems that learn from data in order to make predictions and interferences." (Swamynathan, 2017, p. 53). On a high level, ML can be categorized into four groups (see Figure 3).

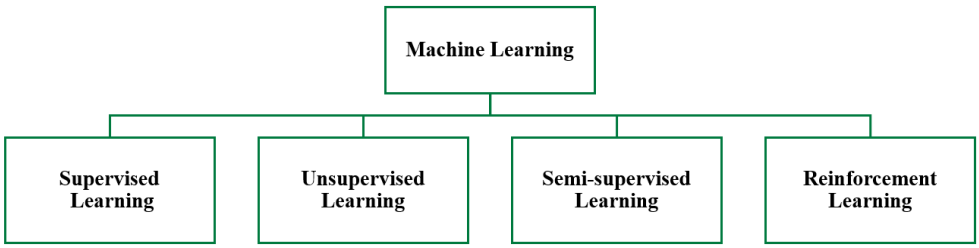


Figure 3: Types of ML (own illustration)

Supervised Learning

Supervised learning (SL) methods represent algorithms that take in a data sample (known as training data) and an associated output (known as label). The main goal of the algorithm is to try to map or associate between the input data sample x and its corresponding output y. The computation is based on multiple training data instances, which, in the future, can be used to predict a new output y' for a new input data sample x' (Sarkar, Bali, & Sharma, 2018, pp. 34–35). This method is called supervised because the algorithm is not only fed with the extracted features of the input data sample, but also with the labels, which represent the correct response. Classification and Regression are two commonly used SL algorithms.

Classification: In classification-based tasks, the objective is to predict output labels that are categorical in nature. Output labels are called *classes* or *class labels*. Based on the desired output, classification problems can either be binary (e. g. sunny or rainy) or multi-class (e. g. classifying colors such as red, blue, green, or yellow). Amongst others, popular classification algorithms include logistic regression, support vector machines, k-nearest neighbors, or decision trees. (Sarkar et al., 2018, pp. 36–37)

Regression: In contrast to classification-based algorithms, regression-based algorithms rely on output responses with continuous numeric values (instead of discrete classes). Regression models use input data attributes (such as neighborhood, m^2, building date) and their corresponding numeric value (e. g. the price of a house) to learn specific relationships between the inputs and outputs and predict outputs for a new, unseen data set. Again, different methods to use regression can be implemented. The most common ones include simple linear regression, multiple regression, or non-linear regression. (Sarkar et al., 2018, pp. 37–38)

Unsupervised Learning

In the case where no pre-labeled data exists and the desired output is unknown for historical data, unsupervised learning (UL) methods are extremely powerful. The UL algorithm aims to learn inherent latent structures, patterns, and relationships from a historical data set without any help regarding output results (Sarkar et al., 2018, p. 38). Thus, UL is more focused on extracting meaningful insights from data, rather than predicting an outcome based on previously available supervised training data. Often, UL is combined with SL technique to build an entire intelligence system with various applications and different outputs. UL methods can be categorized into four broader areas.

Clustering: With clustering, a given data set can be divided into logical groups of related items. Within each group (also called cluster) similarity is high, whereas between each group, similarity is low. This approach is completely unsupervised, since the algorithms tries to detect patterns and put them into groups, without any prior training or supervision. Major approaches of clustering include k-means clustering or the Gaussian mixture model. (Sarkar et al., 2018, pp. 38–39)

Dimension Reduction: Dimension reduction aims to map a large dataset to a lower dimensional space. Each data set is reduced by its features and dimensionality through extracting or selecting a set of principles or representative features. A typical feature reduction would for example result in obtaining a two-dimensional feature space out of a three-dimension structure. (Sarkar et al., 2018, pp. 39–40)

Anomaly Detection: Anomaly detection can also be referred to as outlier detection. Its aim is to find occurrences of rare events or observations that typically do not occur and are thus rare events (Sarkar et al., 2018, p. 41). Mostly, outliers occur infrequently, but can sometimes also happen with a specific pattern over time. An unsupervised algorithm based on a “normal” dataset would then be able to identifiy anormal data points, since they deviate from the trained normal data points. Anomaly detection applications are particularly interesting in the discovery of credit card frauds, network issues or security attacks (Sarkar et al., 2018, p. 41).

Association Rule-Mining: Association rule-mining can be expressed as a data mining method that is used to examine and analyze large transactional datasets to find patterns and rules of interest (Sarkar et al, 2018, p. 41). It is often used, for example, to analyze customer’s shopping patterns due to the ability to correlate products and items.

Semi-supervised Learning

As the name suggests, Semi-supervised Learning (SSL) falls between supervised and unsupervised learning methods. Usually, SSL methods contain aspects from both SL (such as a small amount of pre-labeled data) and UL (such as a lot of training data that is unlabeled) to broaden the limited application spectrum of UL and SL. The usual procedure first uses unsupervised learning algorithms to cluster data and then adds the labeled data via a supervised learning algorithm to label the rest of the unlabeled data. Speech analysis or internet content classification are common use cases of SSL (Gupta, n. d).

Reinforcement Learning

Reinforcement Learning (RL) differs from SL, SSL, and UL, in that its main objective is to map situations to actions that return the maximum final reward (Swamynathan, 2017, p. 69). The algorithm not only considers the immediate reward, but also the next and subsequent. Usually, RL includes an agent that is interacting in a specific environment. By setting and updating policies and strategies, the agent gets a reward for his actions and updates its current strategies or policies if needed. This process continues until gets the most optimal policy. One of the most famous example of reinforcement learning is Google's AlphaGO[1].

2.4.2 Artificial Neural Networks and Deep Learning

Another subgroup of artificial intelligence is the ANN, which builds the basis for deep learning techniques. ANNs consist of simple elements called neurons that form a system to process information or inputs and therefore aims to simulate the functioning of the human brain. The neurons exchange signals between each other through connection links, which can be stronger or weaker and determine how the information is processed (Vasilev et al., 2019, p. 36). Moreover, each neuron has an internal state that is defined by the incoming connections from other neurons as well as an activation function that determines the neuron's output signal (Vasilev et al., 2019, p. 36). By connecting several neurons, layers are built and form the architecture of the ANN. Generally, an ANN can consist of multiple layers, depending on the art of network. However, they always include an input layer, which represents the dataset and the initial condition but does not count as part of the other layers. Furthermore, when using a multi-layer network, additional hidden layer can be inserted that are either connected to another hidden layer or the final output layer, which is represented by y. Every neuron of a layer is connected to every other neuron from the previous and the following layers (Vasilev et al., 2019, pp. 40–41). Furthermore, the layers are activated with a specific activation function such as sigmoid function, the rectified linear unit, or the hyperbolic tangent. They differ depending on the use case and architecture of the neural network. Due do this humanoid architecture, deep learning algorithms based on ANN are able to understand constructs of input examples, recognize the basic characteristic of examples, and make predictions based on those characteristics. Such level of complexity is missing in machine learning algorithm (Vasilev et al., 2019, p. 68). The following figure represents a general structure of a multi-layer artificial neural network.

[1] See Deepmind for more information on the first computer to defeat human GO players.

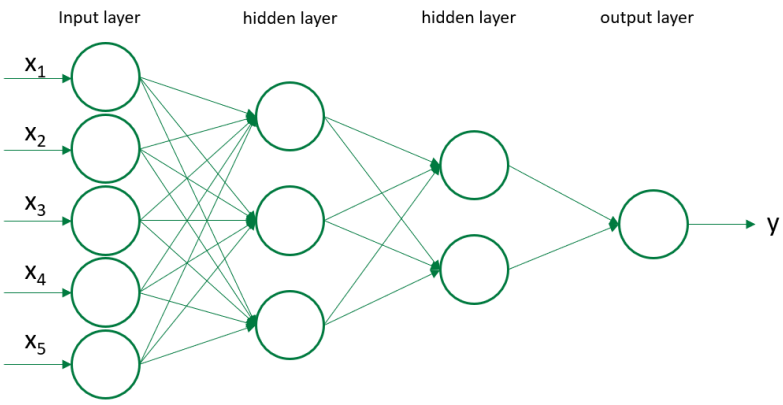


Figure 4: Multi-layer artificial neural network architecture (own illustration)

Just like in ML, there are different architectures or methods to use ANNs. The most common and advanced include Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), Gated Recurrent Network (GRU), and transformers, which are described below. Additionally, a quick detour to transfer learning is provided.

CNN

Originally, CNNs have been proven very successful in image related tasks such as image classification or recognition. They still play a crucial part in enabling vision to self-driving cars or other robots. However, they have become more effective in NLP tasks recently, too, making them an important deep learning technique for several problems. A CNN consists of several types of special layers, that produce outputs such as an n-dimensional map. Generally, a CNN is very similar to the biological cells in the visual cortex of the human brain (Vasilev et al., 2019, p. 73).

RNN

In contrast to the CNN, the RNN is distinguished by its memory of previously processed inputs. Therefore, the RNN process sequential information by combining the latest input sample with the previous state of the network (internal state or memory). Thanks to this ability, RNNs make good use for text or time-series data (Vasilev et al., 2019, p. 73).

LSTM

Even though RNNs are able to retain information, they experience the problem of vanishing or exploding gradients (Vasilev et al., 2019, p. 209), which leads to the problem of holding long-term dependencies in the memory. With their special crafted memory cell though, LSTM are able to handle long-term dependencies and solve the vanishing gradient problem. In contrast to the RNN, the cell state stays constant if there is no outside disturbance since information can only be written in or removed *explicitly* (Hochreiter & Schmidhuber, 1997). This is done by specific gates that allow information to pass in or out (see Figure 5). The first gate, the forget gate, is activated by the sigmoid function which transforms values between 0 and 1. By that, the network can learn which data should be kept (1) or which data

should be forgotten (0). The second gate, the input gate, decides which values will be updated and thus which new information added to the cells by passing the information through the sigmoid function. Alternatively, information is also passed through a hyperbolic tangent function to help regulate the network and calculate new candidates that could be added to this step. Both calculations are combined and the new state updated. The third gate, the output gate, will finally decide what information is included in the output.[2] (Vasilev et al., 2019, pp. 209–212)

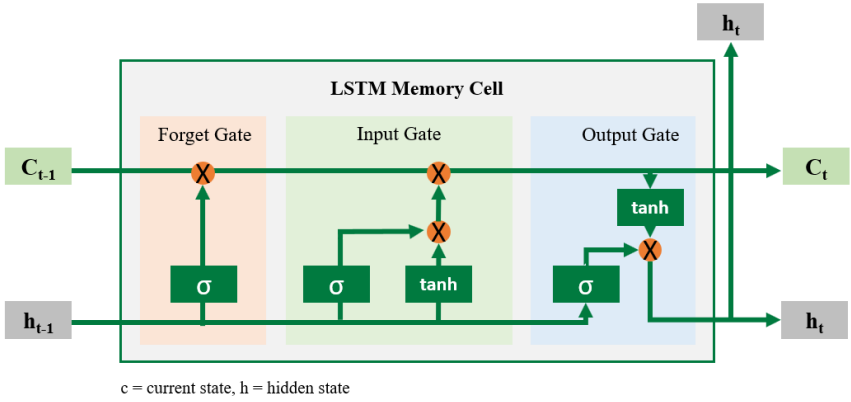


Figure 5: LSTM Memory Cell (own illustration based on Hochreiter & Schmidhuber, 1997)

GRU

The GRU is very similar to the LSTM but includes fewer parameters. It consists of only two gates, the update gate and reset gate. The update gate decides what information will be passed through the cell while the reset gate decides on how much of the previous state is passing through. (Vasilev et al., 2019, pp. 212-214)

Transformers

Despite the rapid advancements of sequence model like LSTM or GRU, there were still a few challenges that impede the use of such models, including the problem of keeping long-term dependencies of sequences and the prevention of parallelization of training data (Vaswani et al., 2017, p. 2). In order to solve this problem and leverage deep learning techniques for sequential data, Vaswani et al. (2017) introduced the concept of transformers. It relies "entirely on self-attention to compute representations of its input and output without using sequence-aligned RNNs or convolution" (Vaswani et al., 2017, p. 2). The model's main architecture can be pictured as in Figure 6.

[2] More detailed information on calculations can be found in Hochreiter & Schmidhuber (1997).

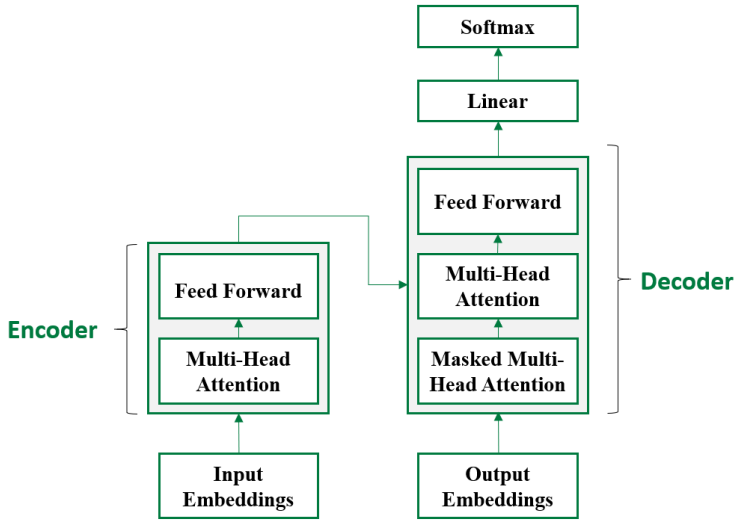


Figure 6: Transformer architecture (own illustration based on Devlin, Chang, Lee, & Toutanova, 2019)

The transformer consists of an encoder and decoder (Devlin, Chang, Lee, & Toutanova, 2019). The encoder block consists of two layers; one multi-head self-attention layer and one feed forward layer. The decoder block includes an additional masked multi-head self-attention layer. Self-attention is a special attention mechanism that allows the model to look at other data in the input sequence to better understand the relation and meaning of a certain word (Vaswani et al., 2017, pp. 3–7). Since self-attention is computed multiple times, it is considered to be a multi-head self-attention. The encoder and decoder are various identical encoders and decoders that are stacked on top of each other. For each sequence, the respective embedded input is passed into the first encoder. The embeddings are transformed and transmitted to the next encoder. From there, the encoded input is passed to the decoder, where it is further processed through the layers of the decoders until it is put through a linear transformation and finally calculated in a softmax function layer (Vaswani et al., 2017, pp. 3–7). Transformers are heavily used in new NLP approaches (Devlin, Chang, Lee, & Toutanova, 2019).

Transfer learning

Originally, ML and DL algorithms were designed to work in isolation. These algorithms were built for very specific tasks and needed to be re-built and re-trained for the adoption of new features. However, transfer learning fixes this obstacle by utilizing knowledge gained from one task to solve a second task. Especially in NLP, this has become a very popular and common approach given the huge time resources to train a model from scratch. Thus, a pre-trained model (such as word embeddings like Word2Vec or GloVe) is used as a starting point for the creation of a new algorithms that solve a related task. Often, transfer learning is seen as a shortcut to save time or reach better results with three main benefits: A higher start in terms of initial skill for the new task, a higher slope in terms of learning curve during the training, or a higher asymptote in terms of a more efficient skill developed during the training (Brownlee, 2017). Figure 7 illustrates the difference between the traditional approach and transfer learning. This research project is based on transfer learning (see chapter 5 for more details).

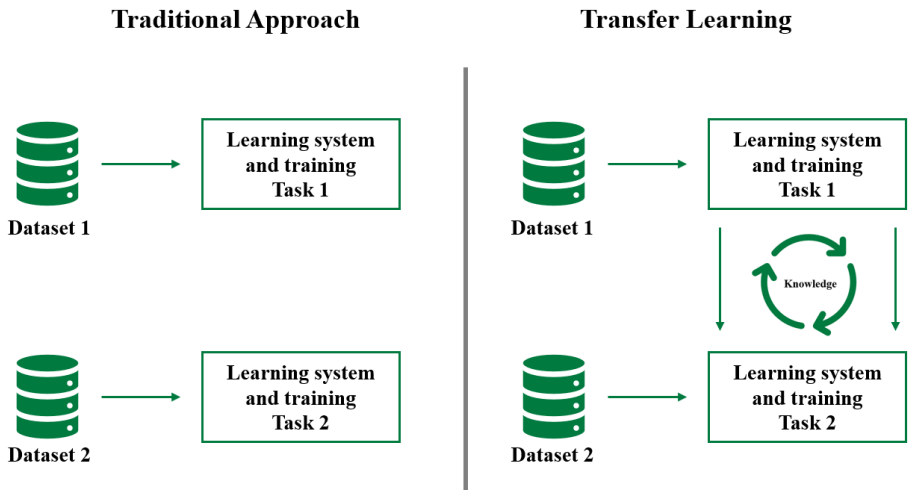


Figure 7: Traditional approach vs. transfer learning (own illustration)

3 RELATED WORK

The following chapter provides an overview of existing research that is relevant to this thesis. Because empathy is an emotion, this chapter includes related works on general emotion recognition in NLP (such as sentiment analysis). Additionally, it describes the most important papers on empathy detection.

3.1 Emotion Recognition

With the rise of AI, research efforts on automatic classification and detection of human emotions gained significant attention in the research community. Other than extracting information from facial or audio information, scientists from NLP and human-computer interaction focus on processing information from textual data. Emotion recognition from text can be done through several approaches such as keyword-based, rule-based, or deep learning based. Within emotion recognition, two research items are of special interest in context of this thesis: *Resources* on emotion recognition and approaches based on *deep learning* for emotion recognition. In the following, only the most important research efforts are mentioned, since an extensive presentation of all results is not focus of this work.

3.1.1 Resources

In the last 30 years of research in emotion modeling there has been a vast number of corpora and lexicons developed, which serve for emotion recognition tasks.

Corpora

The word corpus is derived from the Latin word for "body" and represents a set of data that is used to feed an algorithm. In NLP, a corpus (pl. corpora) is a body of linguistic data in the form of written text.

Already in 1994, the international survey on emotion antecedents and reaction project *(ISEAR)* published a corpus containing 7,666 emotion-labeled sentences (Scherer & Wallbott, 1994). Aprox. 3,000 students reported on various situations in which they experienced the following emotions: joy, fear, anger, sadness, disgust, shame, and guilt (Alswaidan & El Bachir Menair, 2020, p. 2941).

Alm, Roth and Sproat published a corpus *(Tales)* containing 15,302 sentences from 185 children stories. However, all annotators agreed only on 1,270 sentences which are ultimately published. Each sentence was labeled according to the following emotions: neutral, anger-disgust, sadness, fear, happiness, positive surprise, and negative surprise. (Alm, Roth, & Sproat, 2005)

In 2007, Strapparava and Mihalcea published their *SemEval-07* corpus which contains 1,250 instances of news headlines from various sources such as BBC News, CNN or the Google News search engine. Each sentence is assigned a score from 0–100 for each of the following emotions: anger, disgust, fear, joy, sadness, and surprise. (Strapparava & Mihalcea, 2007)

Another corpus build on conversations is *DailyDialogs* (Li et al., 2017). It consists of 13,118 sentences that are manually annotated. Each sentence is labeled according to the six emotions of Ekman (fear,

anger, joy, sadness, disgust, and surprise) (1992), supplemented with "no emotion".

Buechel and Hahn (2017) developed a corpus based on multiple genres and domains *(EmoBank).* It contains 10,548 sentences. Each sentence is annotated twice: according to the emotion expressed by the writer and according to the emotion received by the reader. The emotional model is based on the valence-arousal-dominance model.

EmotionLines published by Chen, Hsu, Kuo, Huang, & Ku (2018) is claimed to be "the first dataset with emotion labels on all utterances in each dialogue only based on their textual content" (p. 1598). Five annotators from Amazon MTurk labeled 2,000 dialogues collected from Friends TV scrips and private Facebook messenger. A total of 29,245 utterances are collected and assigned to one of seven emotions.

Bostan and Klinger (2018) analyzed existing corpora for emotion detection and published a novel unified domain-independent corpus *(Unify Emotion)* which is based on eleven emotions as the common label set.

Recently, Mohammad et al. have published *SemEval-18* and *Sem-Eval-19* (Mohammad, Bravo-Marquez, Salameh, & Kiritchenko, 2018; Chatterjee, Narahari, Joshi, & Agrawal, 2019). The first is based on twitter tweets where each tweet is either neutral or represented by one or more emotions from a choice of eleven emotions. The second contains textual dialogues between to individuals and consists of more than 35,000 instances (Alswaidan & El Bachir Menair, 2020, p. 2942).

Lexicons

A lexicon is a list of words that contains associated information for each word (e. g. if it is a noun, verb, positive, negative, etc.) and is heavily used in NLP to extract meaning from words.

The *WordNet* (Fellbaum, 1998) is a large English database consisting of nouns, verbs, adjectives, and adverbs. They are grouped into a set of cognitive synonyms (called synsets) and interlinked between each other. This results in a network of meaningfully related expressions. The lexicon consists of a total of 117,000 of such synsets.

Strappavara and Valitutti (2004) presented a new lexicon based on WordNet with an affective extension. The so-called *WordNet-Affect* lexicon includes 28 different emotions that are assigned to the emotional categories of positive, negative, ambiguous, and neutral. For example, the positive emotional category includes emotions such as joy or excitement, whereas the negative emotional category contains anger or sadness.

Hu & Liu (2004) developed *Bing Liu* out of approximately 6,800 words from product reviews. Each word is rated either as positive or negative. The final lexicon consists of 2,006 positive and 4,783 negative words. Additionally, it also includes social-media markup, slang, or morphological variants.

The *SentiWordNet* lexicon was particularly designed for opinion mining applications (Esuli & Sebastiani, 2005). It was developed by automatically annotating the WordNet synsets and attaching positive

and negative sentiment scores

In 2011, Nielsen presented the *AFINN* lexicon (Nielsen, 2011). It consists of more than 3,300 English words that are all manually rated on a score from -5 (negative) to +5 (positive) according to the level of valence.

Mohammed, Zhu , & Kiritchenko (2014) developed the *Sentiment140 Lexicon* by extraction more than 1.6 Million tweets. All tweets contain positive or negative emotions. The list contains multiword, including approx.. 62,000 unigrams, 677,700 bigrams and 480,000 pairs tagged as either positive or negative.

The Valence Aware Dictionary and Sentiment Reasoner *(VADER)* is a gold standard list of lexical features and validated by human raters (Hutto & Gilbert, 2015, p. 1). It is based on microblog contents such as social media posts. Each labeled word is combined with five general rules that represent grammatical and syntactical rules for emphasizing the intensity of the given sentiment.

Buechel & Hahn (2018) worked on a novel technique to convert between different emotion formats and developed the emotion representation mapping. With its evaluation on highly multilingual data sets, they could prove that this technique is as reliable as human annotation. Furthermore, the authors created a new emotion lexicon *(EmoMap)* that covers a total of 13 languages. Generally, the words are rated on a scale from 1–5 according to five emotions (joy, anger, sadness, fear and disgust). The datasets contain up to 13,000 words per lexicon.

The Weka[3] machine learning workbench provides a whole package that offers methods for calculating state-of-the-art affect analysis features from tweets (Bravo-Marquez, Frank, Pfahringer & Mohammad, 2019). The package *AffectiveTweet* is a set of programs that is made to analyze emotions and sentiments of tweets. The packaged gained attention since it was used by several winning teams of the SemEval-2018[4] or EmoInt-2017[5] competition.

3.1.2 Deep Learning Approaches

The elaborations about DL mentioned in chapter 2.4.2 clearly show the benefits of using deep neural networks to solve complex problems. Thus, many researchers have been using DL approaches in emotion recognition tasks.

Opposed to traditional methods where sentiment analysis is usually treated as a single-label supervised learning problem, Wang, Feng, Wang, Yu, & Zhang (2016) considered the emotion detection in microblogs (such as tweets) as a multi-label classification problem. To solve this problem, the authors leveraged the skip-gram language model and employed a CNN. According to the authors, the model outperforms strong baselines and accomplishes excellent performance (Wang et al., 2016, p. 567).

[3] Waikato Environment for Knowledge Analysis (find more information on WEKA).
[4] See EmotionIntensity-SharedTask for more information on the competition.
[5] See SemEval-2018 for more information on the competition.

Competing in the SemEval-2017 competition, Baziotis, Pelekis and Doulkeridis (2017) ranked 1st in a subtask about message polarity classification with their implementation of a LSTM augmented with two attention mechanisms. According to the authors, using a more sophisticated version of a general RNN (such as a LSTM or GRU) overcomes the problem of the difficulty to train such RNNs. In their experiment, they achieved slightly better results with the LSTM. Depending on the subtask from the competition, the research team proposed two different kind of models. One model represents a message-level sentiment analysis, whereas the other represents a topic-based sentiment analysis. The first contains a two-layer bidirectional LSTM (Bi-LSTM), the latter a siamese[6] bidirectional LSTM with a distinct attention mechanism than the first.

Ragheb, Azé, Bringay, and Servajean (2019) proposed a model containing encoders and classifiers in order to detect emotion in textual conversations. The encoder is a normal embedding layer where a linear decoder to understand the language model encoder was used and then replaced by the classifier layers. The output of the embedding layer is fed into a three stacked Bi-LSTM trained by average stochastic gradient descent (Ragheb et al., 2019, p. 252). Furthermore, the authors applied self-attention mechanism followed by average pooling. The difference between the pooling from the first and third part of the conversation is then fed into the classifier, represented with two linear layers and a softmax. The authors propose to use the model multi-party conversations and to track emotional changes during long conversations (Ragheb et al., 2019, p. 254)

As a result of their participation on the EmoContext[7] at the SemEval-2019 competition, Basile et al. (2019) proposed four neural systems to detect emotions in English written conversations between a chatbot and humans. Each system's task was to detect sadness, happiness, and anger and split them from the rest (others). The first system is represented by a three-input model, where word embeddings feed a two-layer bidirectional long short-term memory (Bi-LSTM), its hidden state combined by an attention mechanism. The second system, a two-output model, uses a similar architecture, but with a single concatenated input with additional tokens to mark the boundaries (Basile et al., 2019, p. 332). The third system uses a fine-tuned Universal Sentence Encoder in combination with a simple feed-forward network. Lastly, the authors modeled a sentence-pair classification problem using BERT. Since all of these four systems have different advantages and disadvantages and their performance differs depending on the task and emotion, the authors also developed a combinatory model. They used a softmax output probability of each class from all the four models (resulting in 16 features) and trained it on several classifiers (such as naïve Bayes, logistic regression, support vector machine or random forest) (Basile et al., 2019, p. 332). Their best approach, a support vector machine, reached the fourth place at the competition.

[6] Siamese are called the networks that have identical configuration and their weights are linked during training (Baziotis, Pelekis & Doulkeridis, 2017, p. 750).
[7] See EmoContext for more information on the competition.

Huan et al. (2019) explored the emotion recognition ability of BERT, the pre-trained language model. For data input, they used the dataset EmotionLines. Since the context differs between speech-based dialogue (Friends TV scripts) and chat-based dialogues (Facebook Messenger), the authors developed two classification models (FriendsBERT and ChatBERT) by using a three-step approach. First, they used a causal utterance modeling to conserve and refine the emotional information in text. Second, they pretrained the model. Third, they applied weighted balanced warming to tackle the problem of unbalanced emotional labels as fine-tuning of the developed model. They tested their model on 240 dialogues consisting more than 6,800 utterances. The achieved results were highly competitive. (Huang et al., 2019)

Recently, Yin, Meng and Chang (2020) proposed SentiBERT, a transferable transformer-based architecture for compositional sentiment semantics. The model consists of three modules. The first module includes BERT as a backbone to generate contextualized representations of input sentences (Yin et al., 2020, pp. 3696–3697). The second module represents a semantic composition module based on a two-level attention mechanism (attention to tokens and attention to children). The third module deals with phrase and sentence sentiment predictors (Yin et al., 2020, p. 3696). With this approach, the authors underline the high transferability of SentiBERT. For example, SentiBERT not only performs very well on sentiment classification, but can be used for emotion recognition tasks as well.

3.2 Empathy Detection in Natural Language

Most resources or approaches in emotion recognition focus on sentiment analysis or are based on general emotions such as happiness, fear, sadness, disgust, anger, or surprise. However, resources and approaches for empathy detection are rare. This might be due to the complexity of the construct of empathy and its various psychological perspectives (Buechel, Buffone, Slaff, Ungar, & Sedoc, 2018). Moreover, most existing works concerning the detection of empathy focus on spoken dialogue rather than text-based dialogues. Examples include call center applications, where 210 hours of spoken call-center conversations were annotated according to different labels of emotions including empathy (Alam, Danieli, & Riccardi, 2018), or the identification of linguistic and acoustic markers of empathy in counselor empathy retrieved from psychological interventions (Pérez-Rosa, Mihalcea, Resnicow, Singh, & An, 2017). To the best of the author's knowledge, only very few studies focus on the detection and prediction of empathy in natural language text. The following section sheds light on different research contributions (such as corpora, lexica, or models) about empathy detection.

Xiao, Can, Georgiou, Atkins, and Narayanan (2012) generated two datasets from clinical trial studies by college students to label empathy. The first dataset was labeled as either empathic or non-empathic on an utterance level. The second dataset followed the Motivational Interviewing Treatment Integrity (MITI) approach to rate the therapist empathy score per session on a Likert scale from 1–7. Annotations were performed by trained human annotators using audio files and the original transcript of the sessions. In a next step, they used this data to train a maximum likelihood classifier and proposed a group of

lexical features, which was then evaluated through correlation with expert-coded scores. They obtain a correlation of 0.558. (Xiao et al., 2012, pp. 1–4)

Three years later, almost the same group of researches published a study on automated detection of empathy in drug and alcohol counseling via speech and language processing (Xiao, Imel, Georgiou, Atkins, & Narayanan, 2015). This time, the authors used automatic speech recognition to transcribe sessions and deployed speech signal preprocessing together with text-based predictive modeling procedures to finally derive empathy ratings. This computer-derived empathy ratings were evaluated against human-based classifications. The authors conclude that using a combination of speech and language processing methods, satisfying results can be achieved to predict empathy. (Xiao et al., 2015, pp. 1–3).

Khanpour, Caragea, and Biyani (2017) studied empathy in online health communities. Their dataset consists of 225 comments from a lung cancer forum resulting in a total of 1066 messages, and 1066 messages from a breast cancer board. All messages were annotated by two graduate students as either empathic or non-empathic. The dataset was fed to various models, whereas a combination of CNN and LSTM performed best. They also proved that empathetic messages have a positive impact on the patient's sentiment in online health communities. (Khanpour et al., 2017, pp. 246–250)

In their study about modeling empathy and distress in reaction to news stories, Buechel et al. (2018) mentioned the lack of a shared corpus for empathy detection and therefore presented the first publicly available gold standard. They collected 418 articles. These were read by people recruited from a crowd work platform (e. g. Amazon MTurk). The crowd workers were asked to share their feelings in a social media post and rate their empathy and distress level after reading the news article. Ratings were based on six items for empathy and eight items for personal distress, each using a 7-point scale. This annotation approach is considered to be novel, since the scores are derived from the author of the statements and not a third-party. In total, 403 people finished the task which resulted in a final number of messages used in the corpus of 1,860. After completion of the corpus, the authors used the dataset to train different algorithms, with a CNN approach performing best. (Buechel et al, 2018, pp. 4758–4762)

Recently, Sedoc, Buechel, Nachmany, Buffone, and Ungar (2020) created the first-ever publicly available empathy lexicon. In their study, they compared different approaches to learning word ratings from higher-level supervision and used a mixed-level feed forward network to create the lexicon. Their final dataset consists of 9,356 words that are each associated to empathy and distress ratings. Moreover, the authors applied a clustering method and named entity groups to gain further interesting insights into the linguistic field of empathy. (Sedoc et al., 2020, pp. 1657–1666)

4 METHODOLOGY

The research methodology of this thesis is guided by the Design Science Research approach by Hevner, March, Park, & Ram (2004) and Hevner (2007). This chapter gives a general overview about the proposed course of action, as well as detailed information on the various stages and cycles of the DSR approach. Furthermore, the chapter provides guidance on specific models that were used inside the DSR approach, such as the MATTER and MAMA framework for natural language annotation.

4.1 Design Science Research

The majority information systems applications are implemented with the purpose of improving a given status-quo and increasing efficiency and effectiveness. But as certain IT artifact must be artificially created to solve an identified organizational problem, traditional research approaches were no longer valid. Hevner et al. (2004) suggest a design-science paradigm that allows to create and evaluate IT artifacts which emerge from the interaction of people, organizations, and technology (pp. 76–77). To be able to understand, develop and evaluate information systems research, the authors present a conceptual framework consisting of both a behavioral-science and a design-science paradigm. The first investigates phenomena by developing and justifying theories, whereas the latter focuses on building and evaluating artifacts (Hevner et al., 2004, p. 79). Therefore, the goal of behavioral-science research is the truth, and the goal of the design-science research is the utility (Hevner et al., 2004, p. 79). Both approaches are inseparable as the truth informs design and the utility informs theory. In sum, the combination of both approaches leads to new knowledge contributions, which can take on different forms. Most commonly, researchers differ between design theories, frameworks, architectures, design principles, models, methods, constructs, or instantiations (Hevner et al., 2004; March & Smith, 1995). The output of this thesis represents an *instantiation*, since it "operationalizes constructs, models and methods" and is a "realization of an artifact in its environment" (March & Smith, 1995, p. 258). The final tool of this thesis consists of various methods and models (such as the annotation method or the deep learning model) and demonstrates its feasibility and effectiveness embedded in an environment (represented in the pedagogical scenario). Furthermore, Gregor & Hevner (2013) propose a knowledge contribution framework showing the type and degree of contribution of given IT artifacts (see Figure 8).

According to the framework, each contribution can be classified according to its solution domain maturity and its application domain maturity (Gregor & Hevner, 2013, pp. 344–347). A design knowledge contribution can be classified as a *routine design* when both solution domain and problem domain maturity are high. Such contributions apply known solutions to known problems and do not generate a major knowledge contribution. Contributions with high application domain maturity but low solution domain maturity are known as *improvements*. They have developed a new solution to a known problem. *Exaptations* are knowledge contributions with high solution domain maturity but low application domain maturity and usually extend known solutions to new problems by adopting solutions from other

fields. Finally, an *invention* is a "radical breakthrough" (Gregor & Hevner, 2013, p. 345) since it generates a new solution for a new problem.

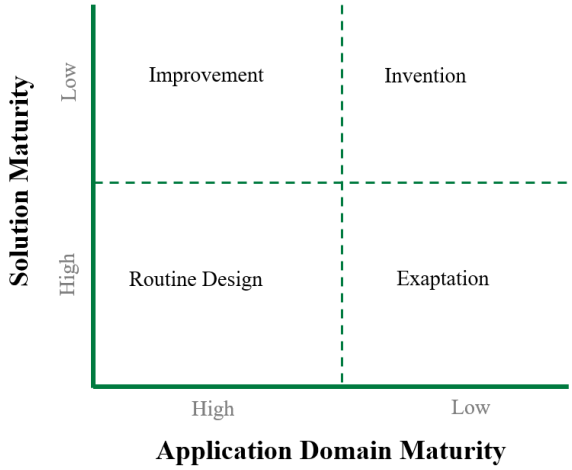


Figure 8: Knowledge Contribution Framework (own illustration based on Gregor & Hevner, 2013)

Overall, the knowledge contribution of this thesis can be classified as an *improvement*. All in all, the problem of decreasing empathy capabilities of students is well-known and existing solutions include coaching, consulting, or empathy courses amongst others. However, this thesis develops a novel approach to address this problem by using state-of-the-art deep neural network techniques to create a better solution that is more efficient, effective, and scalable. But, with regard to the developed corpus that is used to train the algorithm for the final tool, it can be argued that this knowledge contribution represents an *invention*. With the rise of AI and NLP, emotion detection in textual data has received more and more research attention. However, empathy detection is still in its infancy and detecting empathy particularly in textual data remains a novel design problem. To the best of the author's knowledge, the annotation approach used in this thesis to capture emotional and cognitive empathy can be considered as a "radical breakthrough" that entailed "an explanatory search over a complex problem space that requires cognitive skills of curiosity, imagination, creativity, insights and knowledge of multiple realm of inquiry to find a feasible solution" (Gregor & Hevner, 2013, p. 345).

Given this overview about the DSR approach, it is now important to introduce the detailed process steps proposed by Hevner et al. (2004) and Hevner (2007) that were used to conduct this research. Figure 9 shows the conceptual framework of the DSR approach containing three cycles of activities.

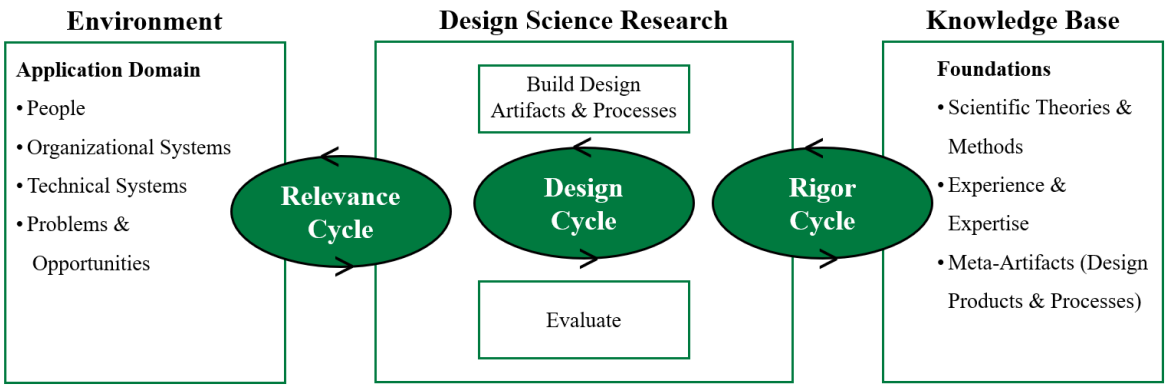


Figure 9: DSR approach (own illustration based on Hevner, 2007)

Relevance Cycle: The first cycle links the environment of the research project with the actual design science activity. The application environment is represented by people, organizational systems, and technical systems, which all aim towards a specified goal. Generally, the goal of design science research is to identify new opportunities to improve practice (Hevner, 2007, p. 3). Therefore, the relevance cycle provides requirements for research projects (such as an opportunity or problem) but also includes criteria for the evaluation of results. At the end of the research, the results will be looped back into the application environment to test and evaluate. This evaluation will ultimately define if more iterations of the relevance cycle are needed to improve its practicability (Hevner, 2007, p. 3).

Rigor Cycle: Following the first cycle, the rigor cycle connects the design science activity to the knowledge base and ensures rigorous research is conducted in the design science process. The knowledge base is built on theories and engineering methods, as well as existing experiences, artifacts, or processes in the specific application environment. According to Hevner, this cycle is crucial to guarantee that the research results add novel knowledge rather than performing routine designs (Hevner, 2007, p. 4).

Design Cycle: Finally, the internal design cycle is the heart of any research project (Hevner, 2007, p. 4). During this cycle, the research activity is iterated through building, evaluating, and refining. The cycle is repeated until a satisfactory outcome is achieved. Both the relevance cycle and the rigor cycle play a crucial role for designing the artifact and must be taken into account during the iterations of the design cycle. Hevner argues that a design activity must be tested in laboratory and experimental setups, before it is evaluated in the application domain (2007, p. 5).

The above-mentioned elaborations regarding the DSR approach yield to seven guidelines that research should follow and that have been followed for this thesis. The first guideline, the *design as an artifact*, requires the creation of a purposeful IT artifact that addresses a vital problem. Secondly, the *problem relevance*, contains the detection of an important and relevant business problem that the IT artifact tries to address. The third guideline includes the *evaluation of the design* to prove its utility, quality, and efficacy. Evaluation can be done observational, analytical, experimental, by testing, or descriptive. The

fourth guideline, the *research contribution*, guarantees that the design artifacts contributes with new knowledge to either the design artifact, the design constructions (e. g. foundations), or the methodologies. The *research rigor*, as described in the fifth guideline, addresses the issue on how the research has been conducted and requires the applications of rigorous methods in constructing and designing the research artifact. Following, the *design as a search process* guideline, includes the creation of effective solutions to the problem space, whereby alternatives are developed and evaluated iteratively. Each iteration follows the cycle approach based on Kuechler & Vaishnavi (2008) consisting of five different steps (see chapter 5 for details on the steps). This work contains three cycles (corpus development, empathy detection with deep neural networks, and the development of the prototype ELEA), which are described in detail in Chapter 5. Ultimately, the seventh guideline concerns the *communication of the research*. This means that the results must be presented to both a technology-oriented and management-oriented audience. (Hevner et al., 2004).

The following table provides an overview of the seven guidelines and their application in this work.

Guideline	**Application**
Design as an Artifact	In course of this research project, three artifacts have been designed and developed: 1) a corpus, 2) deep neural networks to detect empathy, and 3) ELEA, a writing-support system for students.
Problem Relevance	Declining empathy amongst students and rapid progress of today's technology and globalization urge to foster empathy as a key competency to educate change-makers and leaders of tomorrow. However, personal support and feedback require massive resources and are not scalable.
Design Evaluation	Every artifact that has been designed during this research project has been thoroughly tested using appropriate statistical measures or direct feedback from users.
Research Contribution	The overall research contribution can be classified as an improvement. However, the first artifact is considered to be an invention, since it follows a novel annotation approach to capture emotional and cognitive empathy.
Research Rigor	Thorough literature review, application of state-of-the art techniques in annotation and NLP, and user experiments have been used in this research project.
Design as a Search Process	Creation of three artifacts based on the DSR approach proposed by Kuechler & Vaishnavi (2008).
Communication of Research	The research effort is communicated throughout this documentation by addressing both a technology- and management-oriented audience.

Table 4: DSR guidelines for this research project (own table based on Hevner et al., 2004)

4.2 MATTER- and MAMA-cycle for natural language annotation

This research work consists of three parts, each is described by a cycle based on the five steps according to Kuechler & Vaishnavi (2008). The methodology of MATTER and MAMA for natural language annotation proposed by Pustejovsky & Stubbs (2012) (see Figure 10) is used for the first and the second cycles. These are, respectively, corpus development and empathy detection with deep neural networks. The methodology of MATTER and MAMA specifically deals with natural language annotation and its further development.

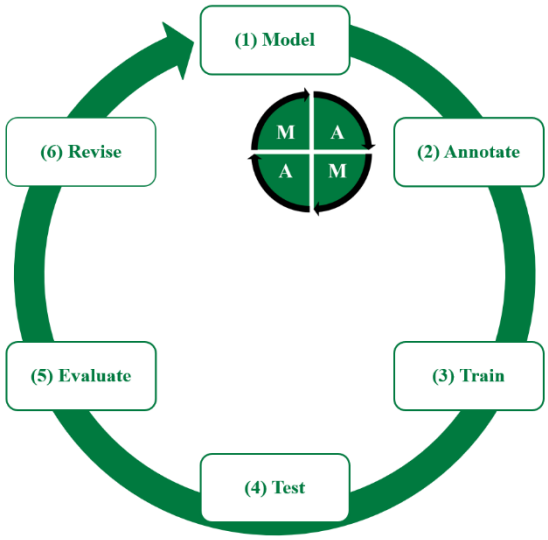


Figure 10: MATTER and MAMA cycle (own illustration based on Pustejovsky & Stubbs, 2012)

Model: The first step consists of creating a conceptual framework for the desired annotation task. This includes various tasks such as gaining familiarity with the topic and searching for background information, defining an annotation scheme, gathering linguistic artifacts, or setting-up a set of annotation guidelines. Usually, the annotation model can be described by the vocabulary of terms (T), the relation between them (R), and their interpretation (I) (Pustejovsky & Stubbs, 2012, p. 25). Thus, the annotation model is described as:

$$M = \{T, R, I\}$$

Annotate: The second step of the process deals with the annotation of the textual artifact itself. For this, human annotators need to be trained and familiarized with the annotation task and the guidelines. Usually, this requires various iterations over modeling and annotation, which is represented by the MAMA cycle (Model-Annotate-Model-Annotate) (Pustejovsky & Stubbs, 2012, pp. 26–28). Each iteration aims to reach high Inter Annotator Agreement (IAA). At the end of this step, a *gold standard corpus* (Pustejovsky & Stubbs, 2012, p. 28), which is a final version of the annotated data that follows the most recent guideline, is created. This final version is used to train the algorithm.

Train, Test: After creating the final corpus, it can be trained and tested with ML (or DL) algorithms. Usually, the annotation corpus is divided into a train set and a test set, with the latter being split up in a development and test part. This is used to judge the performance on the algorithm on unseen data.

Evaluate: Evaluating the utility of the algorithm for the purpose of the annotation task is crucial for further implementations of the algorithm. Utility can be measured differently, most common metrics include accuracy, precision, recall and F1-Score (see chapter 5.2.4 for more details).

Revise: The last step of the MATTER cycle points back to the beginning of the cycle. It includes error analysis or confusion matrices in order to better understand where the algorithm did not perform as intended. This information is used to adjust the model and restart the process to improve performance of the algorithm. (Pustejovsky & Stubbs, 2012, p 31)

5 IMPLEMENTATION

This chapter presents the execution of the methodology described in the prior chapters. Following the DSR approach, the implementation is conducted in three cycles: 1) corpus development, 2) empathy prediction with deep neural networks, and 3) development and testing of ELEA. Each cycle is described with the five-step-approach proposed by Kuechler & Vaishnavi (2008).

Awareness of the problem: The first step focuses on defining the problem and creating awareness. Problem sources may include industry developments, outcomes of other projects or problems from related disciplines. The DSR method requires that the criteria for evaluation and the criteria that signify the end of a cycle are defined.

Suggestion: During the suggestion step different approaches to the problem are investigated. This step is considered to be creative since existing and new elements are combined to propose alternative ideas and approaches.

Development: In this step, the suggestion made in step 2 is now developed and implemented. Depending on the chosen artifact, form and technique for the development and implementation can be diverse.

Evaluation: Once the development phase has been successfully executed, the result will then be evaluated according to the criteria defined in step 1. Evaluation can happen quantitatively or qualitatively and must include explanations. In many cases, deviations from the expectations stated in step 1 lead to another round of suggestions and adaption of the artifact.

Conclusion: At the end of the research effort, results are consolidated and communicated accordingly. Knowledge contribution should be clearly defined. Depending on the result and the expectations, this step is either the end of the complete research project or indicates the start for another research cycle.

5.1 First Design Cycle: Corpus Development

The first design cycle aimed to construct a corpus for modeling empathy in student-written peer reviews. The following section explains in detail how the corpus was created and presents the proposed annotation scheme based on developed annotation guidelines.

5.1.1 Awareness of the problem

With empathy detection being a relatively new field of research, the first step was to gain a deep understanding of current research efforts and investigating existing corpora for modeling empathy. Chapter 3 states the most important research efforts in the field of emotion recognition with a special focus on empathy detection. However, as the literature review shows, publicly available annotated corpora concerning empathy detection in textual data are rare. Either developed corpora are not made available for the public, or they focus on spoken and visual data instead of purely text-based data. Additionally, recently created corpora (such as Buechel et al., 2018) lack the alignment with psychological constructs

and theories of empathy, and do not include precise annotation guidelines. To the best of the author's knowledge, there is also no corpus available that fits the task to train a model that provide students with support in regards of their expressed empathy in common pedagogical scenarios. Therefore, this first cycle focused on developing a corpus for modeling empathy in student-written peer reviews and ended after a final corpus of 500 peer reviews were annotated.

5.1.2 Suggestion

To address the above-mentioned research problem, the construction of a new corpus for modeling empathy in student-written peer reviews was proposed.

As explained in the MATTER-framework above, the first step in creating a corpus is the modeling of the annotation task. This includes a literature review to understand relevant theoretical concepts, defining an annotation scheme, search for or creating linguistic artifacts, choosing an annotation language, and creating annotation guidelines (Pustejovsky & Stubbs, 2012). Creating rigorous annotation guidelines is a crucial step in creating useful corpora. Stab and Gurevych (2014), for example, created a 27-page long annotation guideline that was used to annotate argumentation structures in persuasive essays. Carlile, Gurrapadi, Ke, and Ng (2018) used clear instructions on how to score persuasiveness, eloquence, or relevance on arguments by carefully defining scores from 1–6 or 1–5.

In this work, the same approach was followed. All the relevant information about the model of the annotation task is included in comprehensive annotation guidelines. In a second step, the annotation guidelines were used to annotate the chosen linguistic artifact. After running through the MAMA-cycle, a final version of the corpus was created.

5.1.3 Development

The development of the corpus was conducted in a 4-step process (see Figure 11).

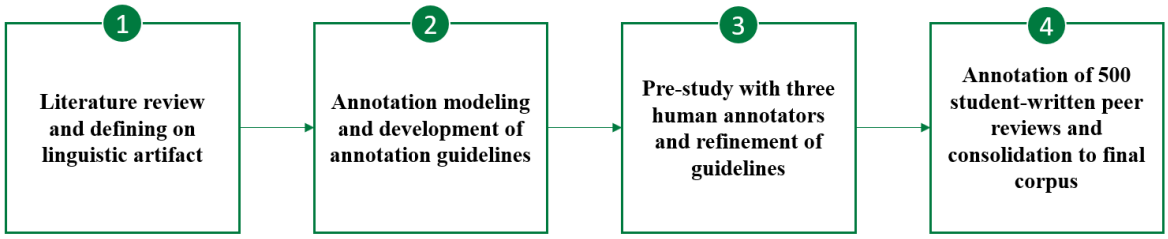


Figure 11: Process of corpus generation (own illustration)

Step 1 and 2 are explained in the following *annotation model* section, step 3 and 4 are explained in the following *annotation* section. The complete annotation guidelines can be found in the appendix (appendix A: Annotation Guidelines). It is a 14-page long document including all necessary details to successfully develop a corpus for modeling empathy in student-written peer reviews.

Annotation model

Before starting to create the annotation guidelines, existing literature on empathy was investigated. The aim of this phase was to gain a deep understanding of the complex construct of empathy and agree on a definition to work with. As stated in the awareness of the problem, annotation tasks should be aligned with psychological terms and theories. An extensive overview about the relevant information on empathy can be found in chapter 2.1. For the sake of the annotation guideline, the information gathered on empathy was filtered, only the relevant content was included in the annotation guidelines. This is enough for the annotators to recognize the construct of empathy and reach a shared understanding. Empathy was therefore defined as the *"ability to react to the observed experiences of another […] and simply understand the other person's perspective "(*Davis 1983, p. 1).

Since the aim of this research project is to create a writing-support system for typical pedagogical scenarios, a suitable linguistic artifact had to be defined. A corpus of 500 student-written peer reviews was collected. Peer reviews are widely used in modern learning scenarios since they enable students to reflect on the content and gain a deeper understanding of it (Rietsche & Söllner, 2019) (also see chapter 2.2). All peer reviews gathered were collected from a mandatory business innovation course of a master's program at the University of St. Gallen. Students were asked to evaluate the strengths and weaknesses of a peer's business model and give suggestions for improvements that would help the peer to adjust the business model. Each student was required to write three of such peer reviews per round. The reviews were part of a final pass/fail rating. All peer reviews were written in German. For the corpus, a subset of 500 peer reviews were randomly collected from around 7,000 documents between 2014–2018.

To generate a corpus that can be used to train a model for a writing-support system, annotations of empathy should not happen on document level, but one level below. This ensures better accuracy and adaptivity of the algorithm when providing feedback for the students. Taking advantage of the general structure of reviews according to Hattie and Timperley (2007) (see chapter 2.2), annotations are based on review *components*. Typically, a review includes three parts in order to answer the questions *"Where I am going and how am I going"* and *"Where to next?"* (Hattie and Timperley, 2007, p. 86): 1) outlining of strengths, 2) outlining of weaknesses, and 3) outlining on suggestions for improvement. Each part is further supported with explanations, details, examples, etc. Consequently, annotations were made per component and included all claims and premises that belong to each certain component. Each component was then given empathy level scores between 1–5.

In summary, Figure 12 illustrates the annotation scheme used for this task.

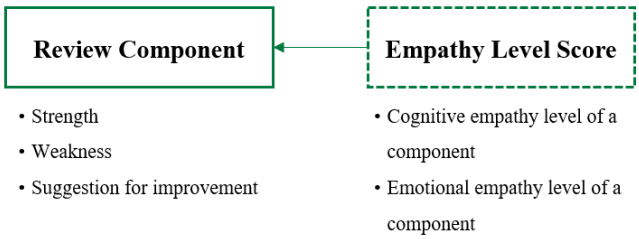


Figure 12: Annotation scheme (based on Wambsganss, Leimeister, Ruckstuhl, Handschuh, & Niklaus, 2020a)

Based on this annotation scheme, the annotation guidelines include further details of how to assess the empathy level score. As defined in chapter 2.1, empathy is the "***ability to react to the observed experiences of another […] and simply understand the other person's perspective*** "(Davis 1983, p. 1). In the context of peer reviews, cognitive empathy refers to the ability of students to set aside their own perspective. A student shows high cognitive empathy if she managed to change her perspective and stepped into the shoes of her fellow student. Additionally, emotional empathy is shown when the student responds emotionally to the peer's affective state and work. A student shows high emotional empathy if she managed to show her own feelings (e.g. in form of excitement, concern, etc.) towards the peer. Because it can be difficult to determine whether a component is empathic or not, the assessment of emotional and cognitive empathy is divided into a 5-score scale, following the examples of Carlile et. al (2018) or MITI (Moyers, Manuel, Miller, & Ernst, 2007). Using the empathy scale from MITI as a basis, both emotional and cognitive empathy scales have been carefully adapted to the selected data domain and described for each score. The following table shows the scale for emotional empathy. The scale for cognitive empathy, as well as original examples and further descriptions for each score can be found in the complete guidelines in appendix A.

	Emotional (affective) empathy
1 = absolutely weak	The student does not respond emotionally to the peer's work at all. He/She does not show his/her feelings towards the peer and writes objectively (e.g. no "I feel", "personally" "I find this.." and no emotions such as "good", "great", "fantastic", "concerned", etc.). Typical examples would be "add a picture." or "the value gap XY is missing."
2 = very weak	Mostly, the student does not respond emotionally to the peer's work. Only very minor and weak emotions or personal emotional statements are integrated. The student writes mostly objectively (e. g. "okay", "this should be added", "the task was done correctly", etc.). In comparison to 1, he/she might be using modal verbs (might, could, etc.) or words to show insecurity in her review (rather, maybe, possibly).
3 = slightly weak / equal	The student *occasionally* includes emotions or personal emotional statements to the peer review. They could be quite strong. However, the student's review is missing personal pronouns ("I", "You") and is mostly written in third person. Emotions can both be positive or negative. Negative emotions can be demonstrated with concern, missing understanding or insecurity (e. g. with modal verbs or words such as rather, perhaps). Typically, scale 3 includes phrases such as "it's important", "the idea is very good", "the idea is comprehensible", "it would make sense", "the task was done very nicely", "It could probably be, that", etc.

4 = Fairly strong	The student was able to respond emotionally to the peer's submitted activity with suitable emotions (positive or negative). He/She returns emotions in his/her review on *various* locations and expresses his/her feelings by using the personal pronoun ("I", "You"). Some sentences might include exclamations marks (!). Typical reviews in this category include phrases such as "I am excited", "this is very good!", "I am impressed by your idea", "I feel concerned about", "I find this very..", "In my opinion", "Unfortunately, I do not understand", "I am very challenged by your submission", "I am missing", "You did a very good job", etc.
5 = strong	The student was able to respond very emotionally to the peer's work and fully represents the affectional state in his/her *entire* review. He/She illustrates this by writing in a very emotional and personal manner and expressing his/her feelings (positive or negative) throughout the review. Strong expressions include exclamation marks (!). Typical reviews in this category include phrases such as "brilliant!", "fantastic", "excellent", "I am totally on the same page as you", "I am very convinced", "personally, I find this very important, too", "I am very unsure", "I find this critical", " I am very sure you feel", "This is compelling for me" etc.

Table 5: Emotional empathy scale

Annotation

After defining the annotation model, the actual annotation process could start. Annotations for this research project were conducted through a web-based annotation tool called TagTog[8]. TagTog allows multi-user annotations and includes both very good graphical user interface and detailed documentation. Furthermore, academic research projects[9] can use TagTog free of charge.

The annotation model in TagTog was defined as follows:

$$M = \{T, R, I\}$$

T = {Component, Strength, Weakness, Suggestion for Improvement, Entity Label, Emotional Empathy, Cognitive Empathy}

R = {Entity Class = [Component: Strength | Weakness | Suggestion for Improvement], [Entity Label: Emotional Empathy | Cognitive Empathy]}

I = {Strength: "Something positive about the peer's submitted work", Weakness: "Something negative about the peer's submitted work, a point of criticism", Suggestion for Improvement: "Something that should be improved or added for the second version of the peer's work", "Emotional Empathy: "1–5", Cognitive Empathy: "1–5"}

Each annotator followed the annotation process defined in the guidelines for each peer review (see Figure 13 for an example). Detailed explanations on how to annotate boarders can be found in the annotation guidelines.

[8] https://www.tagtog.net/
[9] See this thesis' project on TagTog here: https://www.tagtog.net/thiemowa/EmpathyAnnotation/

1. *Reading of the entire peer review:* The annotators are confronted with the student-written peer review and are asked to read the whole document. This helps to get a first impression of the review and to get an overview of the single components and structure of it.
2. *Labeling the components and elaborations:* After reading the entire student-written peer review, the annotator was asked to label the three different components (strengths, weaknesses, and suggestions for improvement). Every supporting sentence (such as explanation, example, etc.) will be annotated together with the according component.
3. *Classification of both empathy scales:* Each component is assessed on its level of cognitive and emotional empathy by giving a score between 1–5.

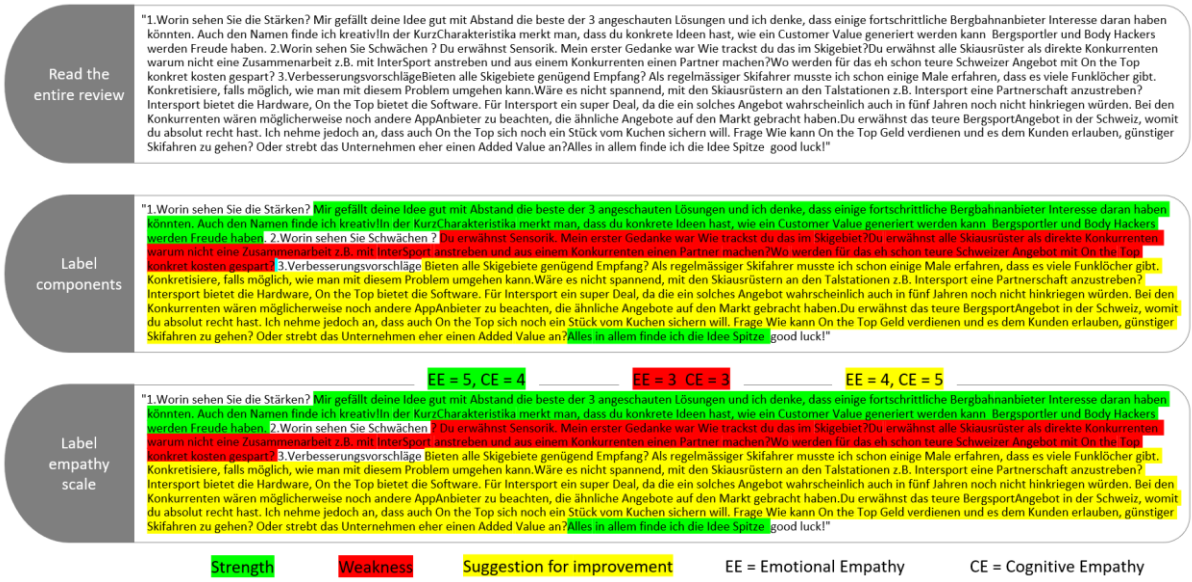


Figure 13: Annotation process per review (own illustration)

Before annotating the entire dataset, a pre-study with three German-speaking annotators from Swiss or German universities was conducted. Eight virtual workshops and several private training meetings were held to train the annotators and test the annotation guidelines on a small sample of data. Possible misunderstandings or ambiguities were discussed, cleared, and in the guidelines adjusted. A subset of 92 peer reviews were annotated by all three annotators in order to evaluate the Inter Annotator Agreement (see chapter 5.1.4). A master version of these reviews was reached by applying the concept of adjunction in TagTog. With adjunction, inconsistencies between multiple annotators are resolved by merging the different version into one final version according to the best-available annotation and agreement between the different annotators.

As satisfying results were obtained after the pre-study, the rest of the dataset was split amongst the three annotators with two annotators evaluating on 130 reviews each and one annotator evaluating on 148 reviews. In sum, the final corpus consists of 500 student-written peer reviews that are annotated according to emotional and cognitive empathy.

5.1.4 Evaluation

Evaluations on a given dataset can differ a lot depending on various agreement studies and notation schema (Meyer, 2014). However, it is crucial to understand the validity, reliability, and agreement of the evaluated dataset (Meyer, 2014). The validity assesses if one can draw conclusions from the data. Its prerequisite is the reliability of the data. The reliability deals with the reproducibility of the data. It assumes that reliability is acceptable if their agreement is good. Finally, agreement can be measured by using different statistical inter-rater agreement coefficients. In this work, the basic percentage of agreement, the chance-corrected Krippendorff's α (Krippendorff, 1980), and Fleiss' Kappa (Fleiss, Paik, & Levin, 2003) were used as statistical inter-rater agreement coefficients. The basic percentage of agreement does not regard agreement by chance or weighted categories (Meyer, 2014, pp. 3–8). However, Krippendorff's α allows further flexibility by permitting arbitrary category metrics and is a widely used IAA metric (Meyer, 2014, p. 18). In contrast, Fleiss' Kappa is used to compare numerical values. Since annotators were also obliged to assess the boundaries of review components[10], Krippendorff's α_U (Krippendorff, 2004) wass also included. This metric demonstrates the differences in markable boundaries. For the evaluation, the data obtained in the pre-study (92 peer reviews annotated from all three annotators) was used, and calculations were based on sentence-level.

Besides calculating the IAA, this chapter also includes results of a disagreement analysis. This was done by creating a confusion probability matrix (CPM) which, in contrast to traditional confusion matrices, is used when more than two annotators are involved in the annotation study (Stab and Gurevych, 2004, p. 632). The CPM includes the conditional probability that an annotator selected a certain category in the column given that another annotator chose a category from the row values (Stab and Gurevych, 2004, p. 632).

First, the agreement of review components was evaluated and second, the agreement of empathy level was evaluated. Evaluation was done using DKPro Agreement[11].

Review components

	Percentage Agreement	**Krippendorff's α**	**Krippendorff's αU**
Strength	0.9641	0.8871	0.5181
Weakness	0.8893	0.7434	0.3109
Suggestions	0.8948	0.6875	0.3512
None	0.9330	0.8312	0.9032

Table 6: IAA review components (based on Wambsganss et al., 2020a)

[10] Details on how to assess those boundaries can be found in appendix A: Annotation Guidelines.
[11] https://dkpro.github.io/dkpro-statistics/.

The obtained results for the Krippendorff's α show almost perfect agreement for the review component *strength* with a score of 0.8871, and substantial agreement for both *weakness* and *suggestions for improvements* with 0.7434 and 0.6875. When looking at Krippendorff's unitizing α, the results are lower. This means that the boundaries of the review components are less precisely annotated as the labels of the review components, especially within the component *weakness* and *suggestion for improvement*. This could be due to the fact that it sometimes hard to mark a clear boundary on what is meant to be a weakness or what it meant to be a suggestion for improvement. Nevertheless, according to Krippendorff's α_U (Krippendorff, 2004), the results still show moderate agreement for the component *strength* and fair agreement for the component *weakness* and *suggestion for improvement*. Moreover, the results obtained clearly show almost perfect agreement for not annotated text spans (0.933 percentage agreement and 0.8312 Krippendorff's α).

When analyzing the CPM between the annotators (see Table 7), the results show a high level of agreement in selecting the different types of review components. The biggest disagreement lies between *weakness* and *suggestions* but is still reasonably high with a value of 0.6065. Again, because weakness and suggestions for improvements are sometimes hard to distinguish and separate, these results are not surprising.

	Strength	**Weakness**	**Suggestions**	**None**
Strength	**0.8340**	0.0347	0.0264	0.1049
Weakness	0.0203	**0.7009**	0.2139	0.0648
Suggestions	0.0214	0.2970	**0.6056**	0.0759
None	0.0742	0.0784	0.0662	**0.7812**

Table 7: CPM of review components (based on Wambsganss et al., 2020a)

Empathy level

With regard to both emotional and cognitive empathy, the Fleiss Kappa indicates moderate agreement[12] between the annotators in both cases, indicating that both empathy levels can reliably be detected in student-written peer reviews. Due to the numerical nature of this annotation task, only Fleiss Kappa has been calculated (Fleiss et al., 2003).

	Fleiss Kappa
Emotional empathy	0.4122
Cognitive Empathy	0.4070

Table 8: IAA empathy level (based on Wambsganss et al., 2020a)

[12] Scores between 0.4 to 0.6 are considered "moderate" (Fleiss et al., 2003)

When interpreting the CPM for the empathy levels, the results show higher disagreement between the annotators compared with the annotation of the review components. However, the results also reveal that the disagreement mostly lays between neighboring scores, meaning that the broader sense of empathy has been captured accurately between the annotators.

	1	2	3	4	5
1	**0.098**	0.440	0.268	0.079	0.066
2	0.137	**0.207**	0.459	0.112	0.027
3	0.053	0.294	**0.316**	0.217	0.063
4	0.024	0.108	0.326	**0.280**	0.208
5	0.044	0.059	0.214	0.470	**0.151**

Table 9: CPM of emotional empathy

	1	2	3	4	5
1	**0.113**	0.343	0.163	0.151	0.155
2	0.114	**0.249**	0.342	0.202	0.035
3	0.024	0.152	**0.207**	0.454	0.106
4	0.013	0.052	0.263	**0.283**	0.327
5	0.022	0.015	0.099	0.530	**0.286**

Table 10: CPM of cognitive empathy

(based on Wambsganss et al., 2020a)

5.1.5 Conclusion

This design cycle contributes to the research by having created rigorous annotation guidelines for empathy detection in peer reviews and presenting the first publicly available corpus for empathy detection in textual data in the educational domain[13].

The first design cycle aimed to create a new corpus for modeling empathy in student-written peer reviews. The suggestion was to construct a corpus that is built on psychological theories and terms, and based on rigorous annotation guidelines. This corpus should be capable of being used to train a model that provides students with support in regard to their empathy expression in common pedagogical scenarios. Before starting to create the corpus, an extensive literature review made sure that the construct of empathy, as used in this work, was fully understood and based on psychological theories and terms. A 14-page annotation guideline included all the details needed to successfully annotate peer reviews according to their empathy levels. The detailed evaluation of a pre-study that was conducted demonstrated the validity and reliability of the dataset and proved the utility of the corpus to train a model.

[13] The corpus will be published pending the acceptance a research paper describing it in Wambsganss, T., Leimeister J. M., Soellner, M., & Ruckstuhl, C. (2020). ELEA: An Adaptive Learning Support System for Empathy Skills.

5.2 Second Design Cycle: Empathy Prediction with Deep Neural Networks

The second design cycle use the artifact created in the first design cycle to create a new one: a model to predict empathy based on state-of-the-art natural language processing techniques. Like the previous chapter, this chapter describes the development of the artifact by the five steps proposed by Kuechler & Vaishnavi (2008).

5.2.1 Awareness of the problem

Empathy prediction is a new field of research and practice. Therefore, specific resources such as corpora are scarce. In order to use the newly created corpus from this research project, new models to suit empathy prediction had to be developed. Natural language processing techniques have already been used to solve similar tasks such as emotion recognition (see chapter 3.1.2). This design cycle therefore aimed to use existing techniques and create a new model for empathy detection which can be used to predict empathy in student-written peer reviews. Since the created corpus from the first cycle included labels that represents the empathy level and since the desired output should exactly be one output label of this multi-class instance, the present task represented a supervised classification task. The cycle ended when satisfactory results are obtained from one of the models, measured in micro F1-score (see chapter 5.2.4 for more details on evaluation metrics).

5.2.2 Suggestion

In the first step of this research cycle, existing techniques for emotion recognition have been analyzed and evaluated. Recent research heavily relies on DL models (such as LSTM, GRU, etc.) rather than traditional ML models (such as regressions, classification trees, etc.) (see chapter 2.4.2). Particularly in NLP related tasks, DL models have outperformed traditional ML algorithms due to their possibility to deal with complex data and characteristics. Furthermore, by choosing a transfer learning approach, the model can benefit from a higher start, higher learning curve and higher performance (see chapter 2.4.2 for more details on transfer learning).

One of the most used and advanced deep neural networks is the LSTM (Khanpour, Caragea, & Biyani, 2017; Li et al., 2019; Buechel et al., 2018; Buechel, Sedoc, Schwartz, & Ungar, 2018). Thus, in order to achieve rigor with this research project, the LSTM network architecture was suggested as a starting point to build the model. Additionally, pre-trained word embeddings are used to get faster and better results. Therefore, Word2Vec, GloVe and Fasttext were taken into consideration.

Recent studies show state-of-the-art accuracy achieved with BERT when dealing with NLP (Devlin et al., 2019; Huang et al., 2019; Yin, Meng, & Chang, 2020). BERT comes with different variations and pre-trained language models, offering a wide range of applications. In 2019, DeepsetAI[14] published a pre-trained German BERT model which significantly outperformed the multilingual models known from BERT, so far, on various downstream tasks. The same research group released FARM, a transfer

[14] https://deepset.ai/

learning Framework for Adapting Representation Models[15]. FARM provides an easy method for adoption of language models and supports simple implementation of transfer learning. This means that prior knowledge or trained models (such as embeddings) can be used and transferred easily to other related use cases, making FARM an interesting framework for this work.

Both approaches (LSTM and BERT) were used and tested in this work. Since empathy was defined as the "*ability to react to the observed experiences of another […] and simply understand the other person's perspective*"(Davis 1983, p. 1) and consists of both emotional and cognitive components, each approach (LSTM and BERT) is used twice: Once in a model predicting emotional empathy and once in a model predicting cognitive empathy.

5.2.3 Development

Before starting to develop the models, data preparation was required. After annotating the data in TagTog during the first design cycle, the data had to be exported, cleaned, and prepared to be used in training. Only after this decisive step could the models be developed and the algorithm be trained on the corpus. The following details provide insights on how these steps were conducted.

All the programming was performed in Google Colaboratory[16] (Colab) since the chosen models required a lot of computational power. Google Colab provides cloud-based GPU processing free of charge and has integrated most deep learning applications such as Keras, Tensorflow, or PyTorch.

The the detail codes and explanations can be found in Appendix B.

Data Preparation

The data was exported from TagTog using a TagTog supplied Application Programming Interface (API). All files were downloaded in two formats: 1) a plain .txt file of the entire peer review and 2) the conducted annotations stored in a .json file. The information from the .json files was electronically read and extracted, converted into standoff data tables, and saved as .ann.txt files. This data format was used to map the annotated text with the original peer reviews. The newly mapped version not only included the annotated parts of the peer reviews, but non-annotated characters as well, labeled with "None". The complete dataset was then transformed to a pandas data frame, which is the usual way to work with data for deep learning models and has additionally been saved to a .csv file (see Appendix B: Source Codes, Data Preparation for the entire code).

Data preparation also included data cleansing. During this research project, a few problems with individual data entries occurred. Either entries were not found (due to e.g. missing annotations caused by human annotator error), or the mapping did not fully work (due to e.g. broken files in TagTog). These errors were manually detected by applying specific filters, resulting in less than 2% broken data. If applicable, the data entries were cleaned manually. If not, the data entries were deleted. In a next step,

[15] https://farm.deepset.ai/
[16] https://colab.research.google.com/

the empathy scores were grouped and renamed to the three labels “non-empathic”, “neutral”, and “empathic”. This ensures better user acceptability and easier classification of empathy for the future writing-support system. The final dataset consisted of 8 columns and 4,174 rows entries. The following figure shows the first few entries of the dataset[17]. More details about the dataset can be found in the appendix (see Appendix C: Data Analysis).

	UniqueID	DocumentID	classID	start	length	f_4	f_5	text
0	0.0	0	None	0	9	None	None	"Stärken
1	1.0	0	strength	9	484	neutral	empathic	Der User Journey Cycle ist vollständig abgebil...
2	2.0	0	None	493	10	None	None	Schwächen
3	3.0	0	weakness	503	390	non-empathic	empathic	Tech Skills sind nicht stimmig dargestellt. We...
4	4.0	0	None	893	15	None	None	Verbesserungen

Figure 14: First entries of the dataset (own illustration)

LSTM

After the dataset has been fully prepared, the LSTM model was built. The first step included loading the dataset and dropping rows that were not needed. In this case, all text entries with a length smaller or equal to three have been dropped. Such text entries contained only whitespaces or single dots and could therefore be ignored for the training. Also, the columns “UniqueID”, “DocumentID”, “start”, and “length” are not needed for this model. Ultimately, the data used for the LSTM consisted of 4 columns and 3,672 data entries. A few data preprocessing steps have been applied, such as removing German stop words or converting to lowercase. Since there is no general rule on how much data pre-processing for textual data should be applied and unnecessary layers could harm the algorithm, the amount of pre-processing has been minimized as far as possible. After pre-processing, the data was split into a training and test set. Because there are two models, the splitting is applied to both the emotional and cognitive empathy label (represented by f_4 and f_5). The splitting is done using the train_test_split function from the natural language library scikit learn[18]. X represents the input data (80% of the data frame; represented by the text), whereas Y represents the output data (20% of the data frame; represented by the empathy labels). By defining a random_state, the splitter maintains the same split point when executing multiple runs. The following code snippet was used to split the dataset:

```
train, test = train_test_split(df, random_state=1, test_size=0.20, shuffle=True)
X_train = np.array(train["text"])
Y_train_f4 = np.array(train["fn_4"]).reshape((-1, 1))
Y_train_f5 = np.array(train["fn_5"]).reshape((-1, 1))
X_test = np.array(test["text"])
Y_test_f4 = np.array(test["fn_4"]).reshape((-1, 1))
Y_test_f5 = np.array(test["fn_5"]).reshape((-1, 1))
print(X_train.shape)
print(X_test.shape)
```

[17] f_4 represents emotional empathy and f_5 cognitive empathy. This is defined by the entity labels in TagTog.
[18] https://scikit-learn.org/stable/

An embedding layer was then added to the model to transform the unstructured textual data to structured vector representations for the algorithm to compute on. First, the output data Y had been transformed to categorized values with the OneHotEncoder. The input data X had been tokenized and padded to the maximum length of 302 characters, which represented the longest input sentence. Tokenization is used to split each sentence into individual pieces, so-called *tokens.* Padding is done to obtain data with the same shape and size. The embedding layer further included a pre-trained language model as explained in chapter 2.4. All three models (Word2Vec, GloVe, and FastText) have been tested, with FastText achieving the best performance. The embedding layer is added to the model architecture (see Appendix B Source Codes, LSTM).

The LSTM's network architecture is based on Google's Keras[19] library (www.keras.io) due to its user friendliness and efficient performance. The following code snipped shows the general architecture of the LSTM model:

```
from tensorflow.keras.layers import BatchNormalization
model_f4 = Sequential()
model_f4.add(
    Embedding(
        input_dim=nb_words,
        output_dim=EMBED_DIM,
        input_length=MAXLEN,
        weights=[embedding_matrix],
        trainable=True,
    )
)
model_f4.add(LSTM (300,return_sequences=True,dropout=0.80))
model_f4.add(Dense(30,activation='tanh'))
model_f4.add(Flatten())
model_f4.add(Dense(20,activation='relu'))
model_f4.add(Dense(4,activation='softmax'))
model_f4.compile(
    loss="categorical_crossentropy",
    optimizer=tf.keras.optimizers.Adam(),#RMSprop(),)
```

After creating an instance of the Sequential class and adding the pre-trained embedding layer to the neural network, the first layer added to the architecture is the LSTM layer with a dropout rate of 0.8. Dropout is the fraction of the units to drop for the linear transformation of the inputs and is defined while fine-tuning the model. Following the LSTM layer, various Dense layers can be added to add more depth to the neural network. Each layer is activated by an activation function, which defines which information is updated or forgotten. The last Dense layer is then used for outputting a desired prediction. After stacking the four layers of the network architecture, the final step before training was the model compilation. During compilation, a loss function is used to find deviations and the optimizer improves the input weights by comparing the loss function and the predictions. The crossentropy loss function computes the crossentropy loss between the predictions and labels[20]. Since this model dealt with more

[19] https://keras.io/

[20] See https://keras.io/api/losses/probabilistic_losses/#categoricalcrossentropy-class for more information on loss functions within Keras.

than two label classes, the categorical crossentropy function was selected. Furthermore, the Adam optimizer was preferred due to its computational efficiency and small memory footprint[21].

With the model architecture defined, the model could be trained and model fitting applied. Model fitting was done in order to prevent overfitting or underfitting of the training data. Different parameters, such as batch size (number of training examples) or number of epochs (amount a dataset is passed forward and backward through the neural network) were defined until the highest evaluation metrics have been obtained. In this research project, satisfactory results were obtained with a batch size of 8 and number of epochs of 3. When working in Colab, a batch size over 8 leads to runtime errors due to high computational resource use. Nevertheless, a batch size of 8 already yields satisfactory results. The final model with the final parameters was then saved using Keras' .save method. The whole process was repeated for cognitive empathy prediction by changing the output label Y to f_5.

BERT/FARM

As suggested, the recently launched BERT architecture with DeepsetAI's German language model was implemented as well. To achieve this, the transfer learning framework FARM was used. The framework provides easy adoption of language models and supports simple implementations of transfer learning. Therefore, the first steps contained the installation of FARM. Then, similar to the LSTM approach, the data was loaded using a pandas data frame and split into a training and test file. Again, the test sample included 20% of all entries from the data frame and a random state was chosen to keep splitting constant during different attempts. The code used to split the dataset with FARM looked like the following:

```
from numpy.random import RandomState
rng = RandomState()
components_train = df.sample(frac=0.8, random_state=42)
components_test = df.loc[~df.index.isin(components_train.index)]
components_train.to_csv('/content/drive/My Drive/Data/Farm/train.tsv',
sep='\t', index=False, header=True)
components_test.to_csv('/content/drive/My Drive/Data/Farm/test.tsv',
sep='\t', index=False, header=True)
```

Generally, FARM consists of three building blocks: the language model, the prediction head, and the adaptive model. The language model is responsible for tokenizing data entries and converting them to vector representations. For this work, the BERT tokenizer loaded with the German language model was used. Furthermore, FARM uses processors to handle conversion from raw text to a PyTorch dataset, where only parameters for conversion need to be defined. The TextClassification processor, which processes the data per sequence, matches this work's downstream task. It was also specified with the multi-label category to be consistent with the four labels from the label_list. 20% of the train file is transferred to a validation file, which is later used for evaluation on unseen data. The maximum sequence length is set to 512 characters which is FARM's limit. However, as sentences in this use case did not exceed this

[21] See https://keras.io/api/optimizers/ for more information on optimizers within Keras.

limit, this was no further issue. The processor was defined as followed:

```
label_list = ['empathic', "non-empathic", "neutral", "None"]
metric = "acc_and_f1"

processor = TextClassificationProcessor(tokenizer=tokenizer,
                                        max_seq_len=512,
                                        data_dir='/Farm',
                                        label_list=label_list,
                                        label_column_name="f_4",
                                        metric=metric,
                                        quote_char='"',
                                        multilabel=True,
                                        train_filename="train.tsv",
                                        dev_filename=None,
                                        test_filename="test.tsv",
                                        dev_split=0.2)
```

Once the processor was initialized, the data was loaded using FARM's DataSilo. The DataSilo stored the train, test and dev dataset and exposed a DataLoader for each set:

```
data_silo = DataSilo(
    processor=processor,
    batch_size=batch_size)
```

The next step represented the architecture of the model. As mentioned before, the model consisted of three building blocks. First, the language model was already defined in the step before. Second, FARM provides various so-called predictions heads which define a certain task the model has to perform. Like the processor, a multi-label prediction head was chosen. Third, all building components were combined in an adaptive model that met the requirements of the downstream task. FARM also provides a built-in optimizer that is added to the model. The following code snippet shows the code used to build the model:

```
language_model = LanguageModel.load(lang_model)
prediction_head = MultiLabelTextClassificationHead(class_weights=data_silo.
calculate_class_weights(task_name="text_classification"),num_labels=len(la-
bel_list))

model = AdaptiveModel(
        language_model=language_model,
        prediction_heads=[prediction_head],
        embeds_dropout_prob=embeds_dropout_prob,
        lm_output_types=["per_sequence"],
        device=device)
model.fit_heads_to_lm()

model, optimizer, lr_schedule = initialize_optimizer(
    model=model,
    device=device,
    learning_rate=learning_rate,
    n_batches=len(data_silo.loaders["train"]),
    n_epochs=n_epochs)
```

Before training the model, the parameters must be fed into the trainer. A trainer permits the management of the entire training process and keeps track of evaluations. This was done with the following commands:

```
trainer = Trainer(
        model=model,
        optimizer=optimizer,
        data_silo=data_silo,
        epochs=n_epochs,
        n_gpu=n_gpu,
        lr_schedule=lr_schedule,
        evaluate_every=evaluate_every,
        device=device)
model = trainer.train()
```

The model was tested with different parameters. As suggested by the authors of BERT, dropout probability was kept at 10% whereas learning rate varied between 2×10^{-5} and 5×10^{-5}, and number of epochs between 2 and 4 (Devlin et al., 2019, Appendix A3). Each combination was evaluated using the metric F1-score (see chapter 5.2.4 for more details on evaluation metrics). The best performance was reached with a learning rate of 3×10^{-5} and number of epochs of 3. The final model was saved and the whole process repeated for the model predicting cognitive empathy.

5.2.4 Evaluation

When evaluating problems concerning classifications (such as text classification, sentiment analysis, etc.), the concept of the confusion matrix is important. The confusion matrix visualizes the model's predictions versus the ground-truth in a tabular way. The row represents the instances of the predicted class, whereas the column represents the instances of the actual class (see Figure 15).

		Actual	
		Positiv	Negative
Predicted	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

Figure 15: Confusion matrix (own illustration)

The confusion matrix helps to understand classification metrics such as accuracy, precision, recall and F1-Score.

Accuracy: Calculates the proportion of true results among the total number of cases analyzed. This performs well when classification labels are well balanced and a quick and easy evaluation is needed. However, accuracy is not able to indicate where a wrong label was predicted and thus makes this metric quite low value.

$$A = \frac{(TP + TN)}{(TP + FP + FN + TN)}$$

Precision: Calculates the proportion of predicted positives that are indeed positive and thus makes the output more precise.

$$P = \frac{TP}{(TP + FP)}$$

Recall: Calculates the proportion of samples from a class that are correctly predicted and thus defines which fraction of actual positives is correctly predicted.

$$R = \frac{TP}{(TP + FN)}$$

F1-Score: Calculates the harmonic mean of precision and recall with equal weight between the two metrics. Thus, the F1-score manages the trade-off between precision and recall, where when precision is rising, recall is dropping and vice versa. In the equation, P is precision and R is recall.

$$\mathrm{F1}_s = 2 \cdot \left(\frac{\mathrm{P} \bullet \mathrm{R}}{\mathrm{P} + \mathrm{R}}\right)$$

In this work, the F1-score was used as the primary evaluation metric. The table below summarizes the micro F1-scores obtained for each model. The micro F1-score aggregates the contributions of all classes to compute the final average. The micro F1-score is preferable in multi-label classification with imbalanced classes (such as the "None" class in this case) (Grandini, Bagli, & Visano, 2020).

Model	(Micro) F1-score
LSTM (emotional empathy)	0.61
LSTM (cognitive empathy)	0.51
BERT (emotional empathy)	0.75
BERT (cognitive empathy)	0.70

Table 11: Overview of F1-Score

The table above clearly shows the higher performance of the transformer model BERT in comparison to the LSTM approach. It is also noticeable that within both approaches, the cognitive empathy model performed worse than the emotional empathy model. This correlates with the results obtained in chapter 5.1.4, where IAA for cognitive empathy was lower than the IAA for emotional empathy. This might loop back to the nature of both constructs: While emotional empathy can be detected by means of emotions, the use of personal pronouns, or certain expressions, it is much harder to find specific patterns

within cognitive empathy. However, the BERT F1-scores of 0.75 and 0.70 respectively still indicated moderate and satisfying results. Moreover, small experiments and tests with the loaded model (see code snippet below) have shown good reliability.

```
basic_texts = [{"text": "Das Template wurde gut umgesetzt Die Darstellung
ist schlüssig, Persona und User Cycle passen zusammen."}]
inferenced_model= Inferencer.load ("/saved_models/cognitiveempathy")
result = inferenced_model.inference_from_dicts(dicts=basic_texts)
PrettyPrinter().pprint(result)
```

5.2.5 Conclusion

The second design cycle successfully created four models,

1. A LSTM to predict emotional empathy
2. A LSTM to predict cognitive empathy
3. A BERT to predict emotional empathy
4. A BERT to predict cognitive empathy.

The models demonstrated desirable performance characteristics for both BERT models which were developed using FARM. Although the Micro F1-scores are lower than usual results in emotion recognition (e. g. in Khanpour, Caragea, & Biyani, 2017; Buechel, Schwartz, & Ungar, 2018), they are sufficient to design a first prototype of a writing-support system for students. Furthermore, the second design cycle contributed to research by creating two well-performing models for empathy prediction of textual data in the educational domain.

5.3 Third Design Cycle: ELEA

After obtaining satisfactory results from the first and second design cycle, the third design cycle marked the last step of this work. The objective of this cycle was to build a prototype of an adaptive writing-support system that is able to detect empathy in student-written peer reviews, provide adaptive feedback, and therefore foster empathy skills amongst students. This chapter thus describes how the prototype ELEA (**E**mpathy **LE**arning **A**pplication) was created.

5.3.1 Awareness of the problem

As mentioned in the introduction of this thesis, empathy amongst university students is in rapid decline. However, UNESCO declared empathy to be a key competency for leaders of tomorrow and included it in the Global Education Agenda 2030. Nevertheless, teaching empathy requires enormous resources. But in new teaching formats such as MOOCs and in traditional didactic formats such as large-group lectures in universities, these resources are limited. The final design cycle confronted these challenges by integrating the artifacts of the previous design cycles to develop a user-centered adaptive tool to support professors in enabling empathy amongst students.

5.3.2 Suggestion

Leveraging the latest technologies in education is an on-going field in research and practice (see chapter 2.3). Particularly in business education, intelligent tutoring systems in the form of adaptive writing-support systems are becoming more prevalent (e. g. Wambsganss et al., 2020b; Chernodub et al., 2019; Lippi & Torroni, 2016). This thesis followed the insights obtained in a very recent report by Wambsganss et al (2020b).

In order to build a user-centered learning tool, Wambsganss et al. (2020b) made use of both top-down and bottom-up approaches. The more rigorous top-down approach concentrated on design meta-requirements from the current state of the literature. This included requirements from educational technology and pedagogical theories (Wambsganss et al., 2020b, pp. 4–5). Inputs from educational technology were used to build the feedback algorithms, based on the latest state-of-the-art NLP techniques. Requirements from pedagogical theories were based on the cognitive dissonance theory (Festinger, 1962), which claims that individual and personal feedback motivates a person to increase learning efforts and improve skills. The more agile bottom-up approach included user testing with low-fidelity prototypes. Summarizing both approaches, seven design principles were derived. These design principles build the foundation of this work's prototype ELEA. The following table summarizes the seven design principles:

#	**Design Principle**
1	Include a learning progress indicator to actively monitor the past and current learning development.
2	Build the tool as a web-based application with a responsive, lean and intuitive user experience.
3	Include a learning dashboard with a choice of different granularity levels to receive the most useful amount of feedback information.
4	Include explanations about the background theory to give student an orientation.
5	Include visual and discourse feedback to receive instant and individual feedback at any time and any place.
6	Include best practices, examples based on theory, and/or how-to guidelines.
7	Provide adaptive and individual feedback to receive useful and specific feedback on their given arguments.

Table 12: Design principles for an adaptive writing-support system (based on Wambsganss et al., 2020b)

5.3.3 Development

Following the second design principle, ELEA was created as a responsive web-based application. It includes a user-centered frontend interface that interacts with the user, and two neural networks that provide feedback on the empathy level in the backend (see Figure 16). The tool's main interface was built in English to make it scalable in the future. However, since the neural networks were trained on German peer reviews, the business model in ELEA is kept in German.

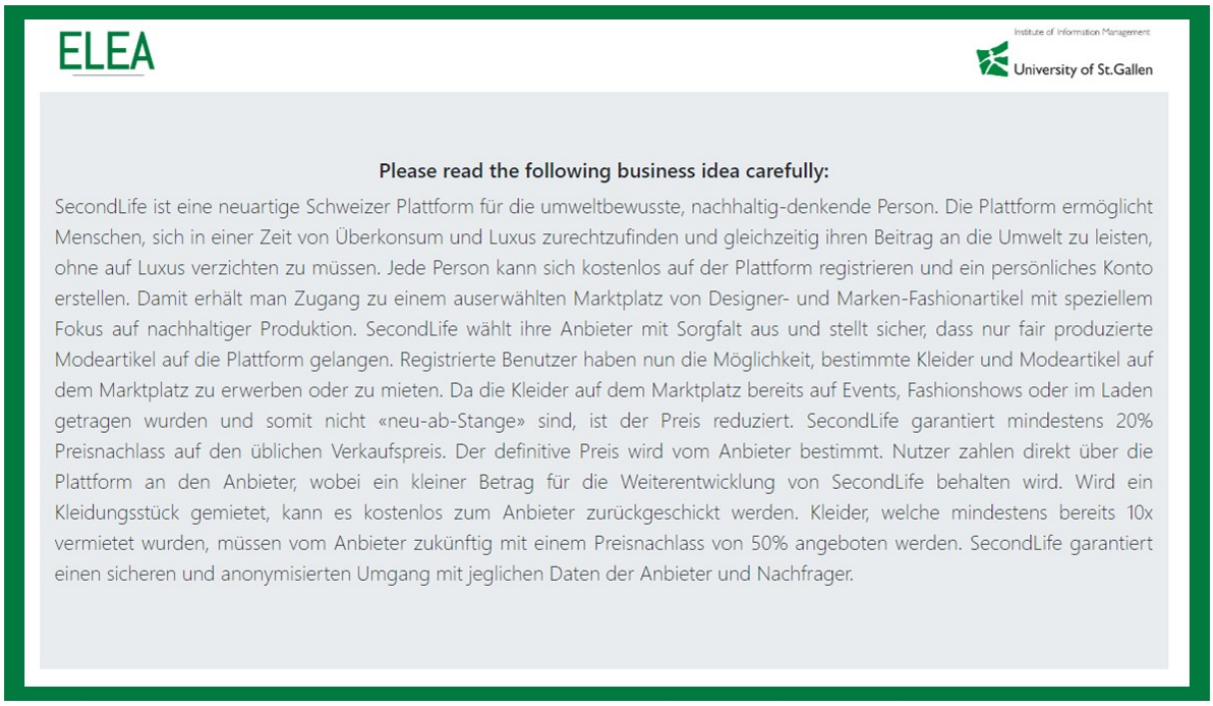
Figure 16: General overview of ELEA (own illustration)

Frontend

ELEA's layout followed a simple, clear, and easy-to-understand user interface. It has been designed using common frontend programming languages html, css and javascript. ELEA consists of four parts: the header, the business model section, the peer review section, and the empathy dashboard. The header contains the logo from ELEA[22] and the university. The business model section includes the business model that the student will read. Just below follows a section with three input fields. The student is given instructions to enter strengths, weaknesses, and suggestions for improvement about the business model presented above. After the student entered her peer review, she can click on the analyze button below. After hitting the button, a new section appears containing the peer's empathy dashboard. The empathy dashboard is divided into two parts with different granularity levels. The first section, the detail discourse feedback, gives insights to each input from both an emotional and cognitive perspective. It also includes examples on how certain inputs could be improved. The second section gives a general overview and provides the student with an illustrative 'overall empathy score' and adaptive feedback on how to improve the peer review. The student can implement ELEA's feedback in her text inputs and analyze the improved peer review again. ELEA will then adapt the empathy dashboard with a new overall empathy score, which allows the student to measure her progress easily. Therefore, design principles 1, 3,

[22] ELEA's logo was created using a free logo maker tool (https://logomakr.com/)

5, 6, and 7 have been implemented in the frontend of ELEA. To include design principle 4, students are able to open a popup window which explains the theory and function behind ELEA. The following illustrations present ELEA's user interface:



Users could get more information on how ELEA works by clicking on a link, that opened the following pop-up window in their browser:

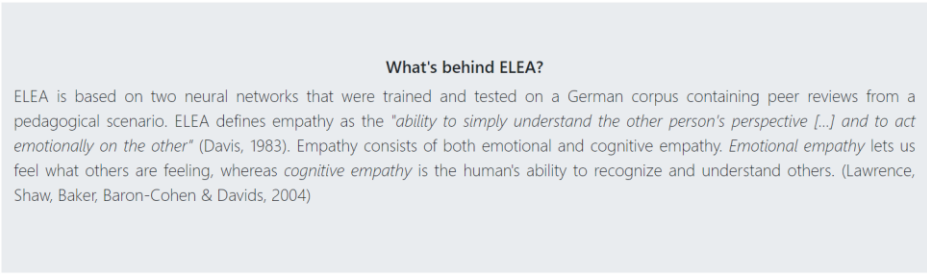


Figure 17:ELEA's user interface (all three pictures; own illustration)

Backend

While the frontend is responsible for the user interface, the backend establishes a connection to the models and the server. For this task, Flask was chosen as the programming language due to personal preference. First, the saved models prom the previous cycles were imported and loaded via FARM's Inferencer. From the model, only the label prediction was needed (see the following code snippet).

```
def inference_emotional(inferencer, basic_texts):
    result = inferencer.inference_from_dicts(dicts=basic_texts)
    label = result[0]['predictions'][0]['label']
    return label.split(',')[0]
```

The second step defined functions that interact with the user interface. Four functions were created (see Appendix B: Sources Codes, ELEA Backend). Two of these functions (once for emotional, and once for cognitive empathy) are responsible to return the empathy level with a small text, according to the detected empathy level form the neural network. Another function calculates the overall empathy score and the fourth function outputs specific text to the user, depending on the total empathy score calculated. The total empathy score is calculated through the summation of all the scores received throughout the detailed feedback, divided by the maximum score possible and multiplied by 100 to obtain a percentage score. Each label "empathic" receives a score of 3, each label "neutral" a score of 2 and each label "non-empathic" a score of 1. Since 6 labels were given in total (3 input fields with each emotional and cognitive empathy), the maximum empathy score to obtain in ELEA is 18. If no label could be detected, no score was given.

$$\text{Total Empathy Score} = \left(\frac{\sum_{n}^{i} \text{Empathy Scores}}{\text{Total Maximum Score}} \right) \times 100$$

The four functions were also integrated to the frontend file to ensure their visibility in ELEA's user interface. This was done by referring to specific variables defined in the flask app (e.g. "emo_labels" in the example below). This dependency was necessary for design principle 4 to fully work. Only the neural networks in the backend were able to analyze the user's input individually and provide adaptive feedback. The following code snipped illustrates the interaction between the frontend and backend:

```
<h4>Detailed feedback on your review</h4>
<br>
  <h5>Strengths</h5>
  {% if 'None' not in emo_labels['strength'] %}
  <p class="text"><i>Emotional Empathy:</i> Your input text was <span
style="color: orange;">{{ emo_labels['strength'].split("'")[1]
}}.</span></p>
  <p class="text text-justify">{{ emo_feedback['strength'][emo_la-
bels['strength'].split("'")[1]] }}</p>
  {% else %}
    No label was predicted. Please re-enter your feedback.<br>
  {% endif %}
  {% if 'None' not in cog_labels['strength'] %}
  <p class="text"><i>Cognitive Empathy:</i> Your input text was <span
style="color: orange;">{{ cog_labels['strength'].split("'")[1]
```

```
}}.</span></p>
  <p class="text text-justify">{{ cog_feedback['strength'][cog_la-
bels['strength'].split("'")[1]] }}</p>
  {% else %}
    No label was predicted. Please re-enter your feedback.<br>
  {% endif %}
```

Moreover, the last section of the Flask app defined the connection between the app, the frontend file (index.html in this case), and the server. The request.method in Flask supports the common HTTP methods such as GET and POST, which were used to request data and send data between the server and the client. Finally, the frontend template needed to be rendered.

```
@app.route('/', methods=['GET', 'POST'])
def index():
    inp1 = {}
    inp2 = {}
    inp3 = {}

    if request.method == 'POST':
        cog_labels = {}
        emo_labels = {}

        cognitive_inferencer= Inferencer.load(model_dir_cog)
        emotional_inferencer= Inferencer.load(model_dir_emo)

        inp1['text'] = request.form['strengths']
        inp2['text'] = request.form['weaknesses']
        inp3['text'] = request.form['suggestions']

        cog_labels['strength'] = inference_cognitive(cognitive_inferencer,
[inp1])
        emo_labels['strength'] = inference_emotional(emotional_inferencer,
[inp1])

        cog_labels['weakness'] = inference_cognitive(cognitive_inferencer,
[inp2])
        emo_labels['weakness'] = inference_emotional(emotional_inferencer,
[inp2])

        cog_labels['suggestion'] = inference_cognitive(cognitive_infer-
encer, [inp3])
        emo_labels['suggestion'] = inference_emotional(emotional_infer-
encer, [inp3])

        percentage_score = calculate_empathy_score(emo_labels, cog_labels)

        feedback = get_feedback(percentage_score)

        del cognitive_inferencer
        del emotional_inferencer


        return render_template('index.html',
                               perc_score=percentage_score,
                               feedback=feedback,
                               emo_labels=emo_labels,
                               cog_labels=cog_labels,
                               emo_feedback=get_emotional_feedback(),
```

```
                                    cog_feedback=get_cognitive_feedback(),)
    return render_template('index.html', onclick = "popup.html")

@app.route('/popup')
def popup():
    return render_template('popup.html'
```

With this application running on the local machine, ELEA is accessible through the localhost on the local machine. However, in order to conduct experiments with users, the application needed to be run on a server which is accessible through a designated address. ELEA was therefore deployed on Microsoft Azure to host it on a cloud server.

5.3.4 Evaluation

To determine if ELEA fulfils its goal to support students in writing empathic peer reviews, online experiments were conducted[23]. Through the Behavioral Lab at University of St. Gallen, 119 participants were recruited. These were randomly assigned to either a 'treatment' group or a 'control' group. In total, 58 participants successfully finished their experiments within the treatment group and 61 participants within the control group. Out of the 119 students, 54 were female. The overall average of age was 24.3 years. The treatment group used ELEA as a tool in the experiment, whereas the control group was using a dictionary-based approach similar to NeuroMessenger (Santos, Colaqo Junior, & Gois de Souza, 2018). The user interface and interaction of both tools were kept the same, only the feedback mechanism changed. The experiment contained three parts: 1) a pre-test, 2) a pedagogical scenario, and 3) a post-test. The pre- and post-test were the same for both groups.

During the pre-test, students were asked eight questions regarding their personal innovativeness and the construct of feedback seeking. These questions helped to assess whether the randomization was successful. However, they were not of further use for the evaluation of ELEA.

For the pedagogical scenario, students were asked to read a business model idea from a fellow peer and provide feedback. They were asked to elaborate on the strengths, weaknesses, and suggestion for improvements, and improve their peer review depending on the results they received from the tool. The treatment group received adaptive feedback from ELEA based on what both neural networks detected when analyzing the input text. The dictionary-based approach from the control group used a list of approx. 25 empathic words to detect them in the user's input (see Appendix B: Source Code, Control Group). However, this approach did not include adaptive feedback. The students did not receive any introduction to the tool prior to the experiment. They were told to enter their inputs in German language.

The post-test included questions from the technology-acceptance model which includes questions about perceived usefulness, intention to use and ease of use (Venkatesh & Bala, 2008). Moreover, the post-test included questions on the perceived level of enjoyment, their perceived empathy skill learning, and

[23] The questions for the students were in German. They have been translated to English for this work.

the perceived feedback accuracy. All questions were measured with a 1–7 Likert scale (1 = "totally disagree", 7 = "totally agree"). At the end of the experiment, students were asked qualitative questions about what they liked or disliked about the tool and provided demographic information. In total, the experiment consisted of 24 questions.

The evaluation of the experiments demonstrates that ELEA is able to support students in writing empathic peer reviews. The results of questions regarding empathy skill learning and perceived feedback accuracy revealed significant differences between both experiment groups (see Figure 18). Students using ELEA as a writing-support system judged their empathy skill learning significantly higher than students using the dictionary-based recommendation system. Moreover, ELEA shows higher perceived feedback accuracy than the control group, which supports the use of individual, adaptive feedback based on artificial neural networks. Figure 18 demonstrates the mean of all participants in their respective experiment group.

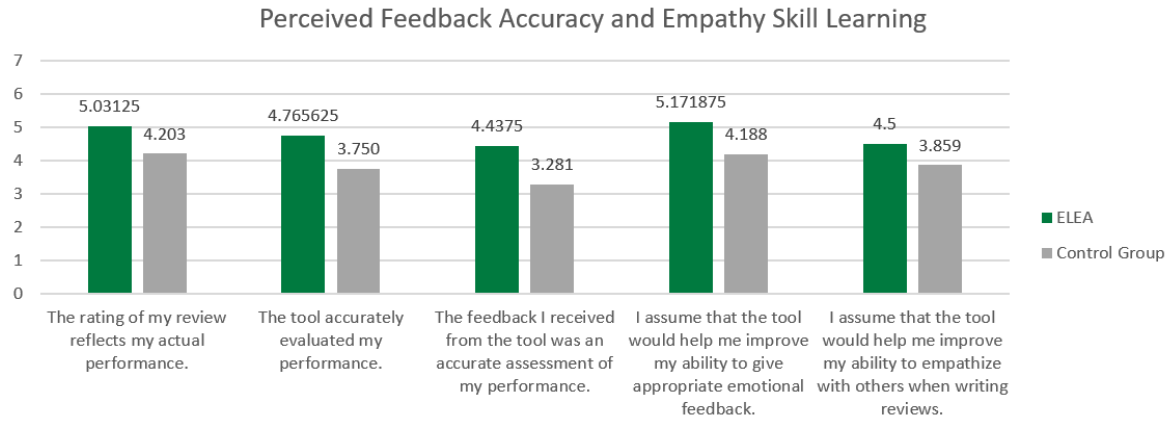


Figure 18: Perceived feedback accuracy and empathy skill learning (own illustration)

Significant differences between both experiment groups could be detected for the technology acceptance constructs, too. Students using ELEA rated their intention to use the tool much higher than students of the control group (see Figure 19). This goes hand in hand with the results obtained from the level of enjoyment (see Figure 20). Moreover, the perceived ease of use shows very high values for both tools, indicating a promising future for both tools in terms of user interaction. Since both tools share the same user interface, the same results for both tools were expected and hoped for.

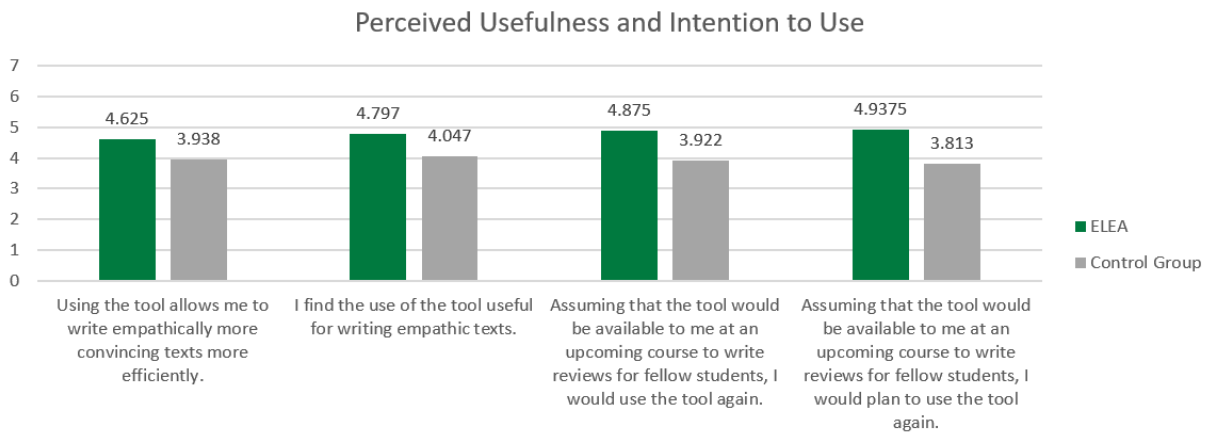


Figure 19: Perceived usefulness and intention to use (own illustration)

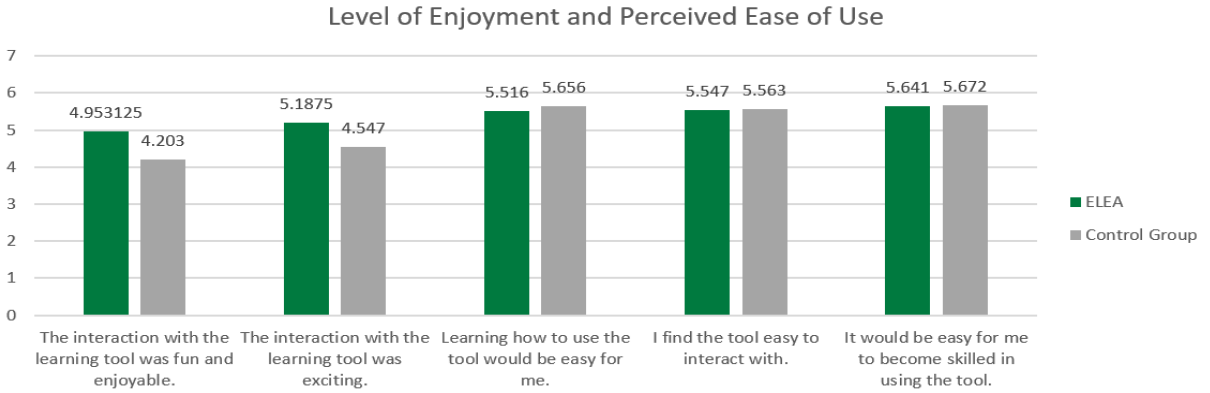


Figure 20: Level of enjoyment and perceived ease of use (own illustration)

The results demonstrate that the students rated the technology acceptance of the adaptive feedback tool ELEA positively compared to the usage of the alternative application. This general positive attitude could also be detected in the students' answers to the open questions at the end of the experiment. However, students also indicated that they would like to receive even more detailed feedback based on more categories or receive concrete text examples on how to improve the empathy score. The table below states some of these answers (translated to English).

#	**Student Answers**
1	*"It was very easy to use and the feedback was helpful!"*
2	*"I liked the fact that it was clearly communicated what kind of empathy is evaluated. The distinction between emotional and cognitive empathy also made it easier for me to accept criticism. In my case I was empathic on a cognitive level, but not on an emotional one. This is also consistent with experiences from my everyday life. I am emphatic, but basically more interested in objective-rational solutions. I think that this tool could help me not only to put myself in the position of a person in terms of content and make suggestions, but also to communicate them better."*

3	*"Exciting new approach, which allows you to gain new insights into your own feedback structure"*
4	*"The tool is very easy to use and does not require any previous technical knowledge. It appeals to a wide range of audiences."*
5	*"I was positively surprised how well the tool could analyze my text. I think such a tool could help many people."*
6	*"It worked very easily and the results appeared very quickly. I liked that it showed an overall score. In addition to the feedback it shows how good the text was and how much potential for improvement there is. I liked that the results were divided into emotional and cognitive empathy, explaining what was missing to improve it."*
7	*"The feedback from the tool came immediately and fitted in well with my answers. Through the tool I noticed where I could have chosen a better formulation. The Empathy Score is a good summary of the feedback and helped me to better assess my answers."*
8	*"I particularly liked the fact that the tool already questioned my own writing style and that the use of the tool did not require any additional effort."*
9	*"It was helpful to distinguish between the two categories of empathy. This again clearly showed me that I do not show emotional empathy enough. It was also useful that the tool said how to show emotional empathy (feelings when reading the business idea etc.)."*
10	*"It could include concrete examples of what exactly is meant by feedback. It could either make improvements directly in the text or at least attach a concrete example below".*

Table 13: Answers to open questions

5.3.5 Conclusion

In the third design cycle a prototype to help students improve their empathy skills in peer reviews was created. The seven design principles, according to Wambsganss et al. (2020), were used to create the prototype as a web-based application. The prototype was well received by the target audience. ELEA is easy-to-use, user-centered and provides students with individual, adaptive feedback on their German peer reviews. It includes adaptive feedback, both in a discourse and illustrative manner. Students are able to improve their peer review according to the feedback ELEA provides and they can also track their progress. ELEA demonstrated a short-term positive influence on the perceived emotional and cognitive empathy skills of students.

All in all, Table 14 provides a recap of the three design cycles conducted in this research project.

Guideline	**Cycle 1**	**Cycle 2**	**Cycle 3**
Awareness of the problem	Literature review, corpus analysis	Conclusion of cycle 1, literature review, analysis of existing models	Conclusion of cycle 2, research of existing tools
Suggestion	Design rigorous annotation guidelines to develop a corpus to detect empathy in German student-written peer reviews	Develop neural network models to predict emotional and cognitive empathy	Create a prototype that supports students to improve their empathy in German peer reviews
Development	14-page annotation guidelines, German corpus to model empathy in German peer reviews	LSTM and BERT models for emotional and cognitive empathy prediction	ELEA, an adaptive writing-support system for students
Evaluation	Evaluation of inter annotation agreement	Model testing, evaluation of performance	Evaluation of user experiments in comparison to a control group
Conclusion	Completion of final corpus and annotation guidelines	Saving of final models for third design cycle	Completion of ELEA and final conclusion

Table 14: Overview of design cycles

6 CONCLUSION

The rapid progress of technology and globalization drives the need for change-makers, thought leaders and team players. They require collaborative competencies like empathy and empathic leadership. But despite the need to foster empathy in education (Kaitlin & Konrath, 2019), students are suffering from a decrease in capacity for empathy development. Potential reasons are learning formats such as MOOCs or large-scale lectures in educational institutions, where educators may not be able to provide students with continuous support and individual feedback throughout their learning journey.

6.1 Summary of Research Questions

This thesis was concerned with the development of a writing-support system that automatically detects empathy in natural language text and provides adaptive feedback to students. Using the design science approach by Hevner et al. (2004), the web-based empathy learning application ELEA was developed. Both scientific literature review (rigor) and the practical application domain such as the organizational system (relevance) defined the starting point. For this thesis, empathy was defined as "***ability to react to the observed experiences of another […] and simply understand the other person's perspective***"(Davis 1983, p. 1), and consists of both emotional and cognitive components. Furthermore, this thesis focused on a pedagogical scenario from a Swiss university which is based on German student-written business models and peer reviews.

Three design cycles were used to build and evaluate the final artifact. Each design cycle aimed to answer one research question. The first research question focused on developing a corpus for modeling empathy in German student-written peer reviews. A 14-page long annotation guidelines was designed in order to annotate 500 peer reviews and create the first publicly available corpus for empathy detection in textual data in the educational domain. The second design cycle aimed to investigate how artificial intelligence can be used to detect and predict empathy in German student-written peer reviews. The solution lied in the use of deep neural networks and state-of-the-art techniques from natural language processing. Bidirectional encoder representations from transformers loaded with a German language model and modeled in a framework for adapting representation models were finally used to detect and predict empathy in German student-written peer reviews. Ultimately, the third design cycle developed the web-based writing-support system ELEA. Final evaluations have shown that ELEA leads to high empathy skill learning and shows very high technology acceptance among students. ELEA manages to directly confront students with empathy and provide them with continuous and adaptive learning feedback regarding their empathy skills.

6.2 Limitations

A few limitations of this work need to be acknowledged. One limitation is represented by the complexity of the construct of empathy itself. Studies suggest that empathy is differently assessed and expressed between genders due to neural distinctions in the human brain (Rueckert & Naybar, 2008). Even though both genders were represented for the annotation of the corpus, which served as a ground-truth for the training of the algorithm, no further considerations were made during the creation and evaluation of the models and ELEA. The complexity of empathy and gender differences is suggested to be focus of a future research project. Another limitation is the relatively small sample size provided by the corpus. More data would potentially lead to more insights in modeling empathy based on textual data. However, for this research project, a sample of 500 peer reviews was enough to create a first prototype. Evaluation on both the corpus and the models proved the suitability of the sample. Ultimately, this work only focused on the very specific knowledge domain of German peer reviews in a pedagogical scenario, which leads to the following suggestions for future research.

6.3 Future Research

This research project is focused only on German peer reviews from a lecture class at a Swiss university. Future research could include the development of a corpus, algorithms and a prototype to detect and predict empathy in another language (e. g. English), for another use case (e. g. persuasive essays), or in another domain (e. g. business, industry domain).

Furthermore, future research could focus on a broader and larger application of the corpus development. This means that more data needs to be annotated, potential adaptions of the annotation guidelines need to be considered, and further annotation studies should be conducted. This also includes further investigations in gender differences when assessing empathy.

Finally, the inputs from the students during the experiment indicate further improvements regarding ELEA's functionalities. For example, ELEA could be further enhanced by adding more empathy dimensions and more concrete examples on how the peer review could be improved to show more emotional and cognitive empathy. Additionally, more explanations on how ELEA works was wished. This could be done by adding multimedia elements such as illustrations or videos, explaining empathy and ELEA. Lastly, ELEA could be implemented as a long-term tool during the student's education. This means that progress could be saved, tracked, or even compared to fellow peers over a longer period.

6.4 Personal Conclusion and Challenges

Personally, this research produced a great deal of uncertainty and many challenges for me. With relatively little experience in programming, I was unsure if and how I would be able to conduct this research project. I am proud to have successfully mastered this challenge and to learned from every aspect of the master's thesis beyond my own limitations. Over and over I was confronted with new challenges: Open-source systems that are under development and did not run smoothly, required dependencies for programming tasks, or installation problems due to incompatibilities with my operating system. Nevertheless, I was able to successfully find solutions and complete this work. I would like to thank my supervisor, Thiemo Wambsganss, for his patience and academic support during the development of this thesis. I would also like to thank all my friends, members of my family, as well as my boyfriend for always having a listening ear for my challenges and for providing me with advice and support.

7 REFERENCES

Aggarwal, C. (2015). *Data Mining. The Textbook.* Cham, Switzerland: Springer International Publishing.

Alam, F., Danieli, M., & Riccardi, G. (2018). Annotating and modeling empathy in spoken conversations. *Computer Speech and Language*. doi:10.1016/j.csl.2017.12.003

Alavi, M., & Leidner, D. E. (2001). Review: Knowledge Management and Knowledge. *MIS quarterly, 25*(1), 107.

Alm, C. O., Roth, D., & Sproat, R. (2005). Emotions from text: machine learning for text-based emotion prediction. *Proceedings of the conference on human language technology and empirical methods in natural language processing* (pp. 579–586). Stroudsburg, PA: Association for Computational Linguistics.

Alswaidan, N., & El Bachir Menair, M. (2020). A survey of state-of-the-art approaches for emotion recognition in text. *Knowledge and Information Systems volume, 62*, 2937–2987.

Bailenson, J. N., Yee, N., Blascovich, J., Beall, A. C., Lundblad, N., & Jin, M. (2008). The Use of Immersive Virtual Reality in the Learning Sciences: Digital Transformations of Teachers, Students, and Social Context. *Journal of the Learning Sciences*, 102–141.

Barnett, G., & Mann, R. E. (2013). Empathy deficits and sexual offending: A model of obstacles to empathy. *Aggression and Violent Behavior, 18*, 228-239. doi:10.1016/j.avb.2012.11.010

Baron-Cohen, S., & Wheelwright, S. (2004). The empathy quotient: An investigation of adults with asperger syndrome or high functioning autism, and normal sex differences. *Journal of Autism and Developmental Disorders, 34*(2), 163–175. doi:10.1023/B:JADD.0000022607.19833.00

Basile, A., Franco-Salvador, M., Pawar, N., Štajner, S., Chinea Rios, M., & Benajiba, Y. (2019). SymantoResearch at SemEval-2019 task 3: combined neural models for emotion classification in human-chatbot conversations. *Proceedings of the 13th international workshop on semantic evaluation* (pp. 330–334). Minneapolis: Association for Computational Linguistics.

Batson, C. D. (2011). These things called empathy: Eight related but distinct phenomena. In J. Decety, & W. Ickes, *The social neuroscience of empathy* (pp. 3–16). Cambridge, MA: MIT Press.

Batson, D. C., Sager, K., Garst, E., Kang, M., Rubchinsky, K., & Dawson , K. (1997). Is empathy-induced helping due to self–other merging? *Journal of Personality and Social Psychology,, 73*(3), 495-509. doi:10.1037/0022-3514.73.3.495

Baziotis, C., Pelekis, N., & Doulkeridis, C. (2017). DataStories at SemEval-2017 Task 4: Deep LSTM with Attention for. *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)* (pp. 747–754). Vancouver, Canada: Association for Computational Linguistics.

Bergemann, E. (2009). Exploring psychotherapist empathic attunement from a psychoneurobiological perspective: Is empathy enhanced by yoga and meditation? California: Pacifica Graduate Institute.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics* (pp. 135–146). Facebook AI Research. doi:10.1162/tacl_a_00051

Bostan, L. A., & Klinger, R. (2018). A survey on annotated data sets for emotion classification in text. *Proceedings of COLING 2018, the 27th International Conference on Computational Linguistics.* Santa Fe, USA.

Bravo-Marquez, F., Frank, E., Pfahringer, B., & Mohammad, S. M. (2019). AffectiveTweets: a Weka Package for Analyzing Affect. *Journal of Machine Learning Research, 20*, 1–6.

Brownlee, J. (2017, December 20). A Gentle Introduction to Transfer Learning for Deep Learning. (D.

L. Vision, Ed.) Machine Learning Mastery. Retrieved from https://machinelearningmastery.com/transfer-learning-for-deep-learning/

Buechel, S., & Hahn, U. (2017). EmoBank: Studying the Impact of Annotation Perspective and Representation Format on Dimensional Emotion Analysis. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers* (pp. 578–585). Valencia, Spain: Association for Computational Linguistics.

Buechel, S., & Hahn, U. (2018). Emotion Representation Mapping for Automatic Lexicon Construction (Mostly) Performs on Human Level. *Proceedings of the 27th International Conference on Computational Linguistics* (pp. 2892–2904). Santa Fe, New Mexico, USA: Association for Computational Linguistics.

Buechel, S., Buffone, A., Slaff, B., Ungar, L., & Sedoc, J. (2018). Modeling Empathy and Distress in Reaction to News Stories. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, (S. 4758–4765). Brussels, Belgium.

Buechel, S., Sedoc, J., Schwartz, H. A., & Ungar, L. (October 2018). Learning Neural Emotion Analysis from 100 Observations: The Surprising Effectiveness of Pre-Trained Word Representations.

Butler, D. L., & Winnie, P. H. (1995). Feedback and Self-Regulated Learning: A Theoretical Synthesis. *Review of Educational Research, 65*(3), 245–281. doi: 10.2307/1170684

Carlile, W., Gurrapadi, N., Ke, Z., & Ng, V. (2018). Give Me More Feedback: Annotating Argument Persuasiveness and Related Attributes in Student Essays. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 621–631). Melbourne, Australia: Association for Computational Linguistics. doi:10.18653/v1/P18-1058

Chatterjee, A., Narahari, K. N., Joshi , M., & Agrawal, P. (2019). Semeval-2019 task 3: emocontext: contextual emotion detection in text. *Proceedings of the 13th international workshop on semantic evaluation* (pp. 39–48). Minneapolis, Minnesota, USA: Association for Computational Linguistics.

Chen, S.-Y., Hsu, C.-C., Kuo, C.-C., Huang, T.-H., & Ku, L.-W. (2018). EmotionLines: An Emotion Corpus of Multi-Party Conversations. *LREC 2018 - 11th International Conference on Language*, (pp. 1597–1601).

Chernodub, A., Oliynyk, O., Heidenreich, P., Bondarenko, A., Hagen, M., Biemann, C., & Panchenko, A. (2019). TARGER: Neural Argument Mining at Your Fingertips. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations* (pp. 195–200). Florence, Italy: Association for Computational Linguistics.

Cuff, B. M., Taylor, L., Brown, S., & Howat, D. (2016). Empathy. A Review of the Concept. *Emotion Review, 8*(2), 144-153. doi: 10.1177/1754073914558466

Davis, M. H. (1980). A multidimensional approach to individual differences in empathy. *JSAS Catalog of Selected Documents in Psychology, 10*, 85.

Davis, M. H. (1983). Measuring individual differences in empathy: Evidence for a multidimensional approach. *Journal of Personality and Social Psychology, 44*(1), 113–126. doi:10.1037/0022-3514.44.1.113

De Vignemont, F., & Singer, T. (2006). The empathic brain: how, when and why? *Trends in Cognitive Sciences, 10*(10), 435-441. doi:10.1016/j.tics.2006.08.008

Decety, J., & Jackson, P. L. (2004). The Functional Architecture of Human Empathy. *Behavioral and Cognitive Neuroscience Reviews, 3*(2), 71-100. doi:10.1177/1534582304267187

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Tech-*

nologies, Volume 1 (Long and Short Papers) (S. 4171–4186). Minneapolis, Minnesot: Association for Computational Linguistics. doi:10.18653/v1/N19-1423

Eisenberg, N. (2000). Emotion, Regulation, and Moral Development. *Annual Review of Psychology, 55*(1), 665-697. doi:10.1146/annurev.psych.51.1.665

Eisenberg, N., Shea, C. L., Carlo, G., & Knight, G. (1991). Empathy-related responding and cognition: A "chicken and the egg" dilemma. In W. Kurtines, & J. Gewirtz, *Handbook of moral behavior and development* (S. 63-88). Hillsdale, NJ: Erlbaum.

Ekman, P. (1992). An Argument for Basic Emotions. *COGNITION AND EMOTION, 6*(3), 169–200.

Esuli, A., & Sebastiani, F. (2005). Sentiwordnet: a publicly available lexical resource for opinion mining. *Proceedings of the 14th ACM international conference on information and knowledge management, CIKM'05*, (pp. 417–422). New York.

Fellbaum, C. (1998). *WordNet: an electronic lexical database. Language, speech, and communication.* Cambridge: MIT Press.

Festinger, L. (1962). Cognitive Dissonance. *Scientific American, 207*(4), pp. 93–106.

Fleiss, J. L., Paik, M. C., & Levin, B. (2003). *Statistical Methods for Rates and Proportions.* (Vol. 3). John Wiley; Sons, Inc.

Gini, G., Albiero, P., Benelli, B., & Altoè, G. (2007). Does empathy predict adolescents' bullying and defending behavior? *Aggressive Behavior, 33*(5), 467–476. doi:10.1002/ab.20204

Goetz, J. L., Keltner, D., & Simon-Thomas, E. (2010). Compassion: An evolutionary analysis and empirical review. *Psychological Bulletin,, 136*(3), 351-374. doi:10.1037/a0018807

Goldman, A. I. (1993). Ethics and Cognitive Science. *Ethics, 103*(2), 337-360. Retrieved from https://www.jstor.org/stable/2381527

Grandini, M., Bagli, E., & Visani, G. (2020). Metrics for Multi-Class Classification: An Overview.

Gregor, S., & Hevner, A. R. (2013). Positioning and Presenting Design Science Research for Maximum Impact. *MIS Quarterly, 37*(2), 337-356. doi:10.25300/MISQ/2013/37.2.01

Gupta, A. (n. d.). *GeeksforGeeks*. Retrieved from Semi-Supervised learning: https://www.geeksforgeeks.org/ml-semi-supervised-learning/

Gupta, S., & Bostrom, R. (2013). An investigation of the appropriation of technology-mediated training methods incorporating enactive and collaborative learning. *Information Systems Research, 24*(2), 454–469. doi:10.1287/isre.1120.0433

Gupta, S., & Bostrom, R. P. (2009). Technology-mediated learning: A comprehensive theoretical. *Journal of the Association for Information Systems, 10*(9), 686.

Gupta, S., Bostrom, R. P., & Huber, M. (2010). End-user training methods: What we know, need. *ACM SIGMIS Database, 41*(4), 9–39.

Hattie, J., & Timperley, H. (2007). The Power of Feedback. *Review of Educational Research, 77*(1), 81-112. doi:10.3102/003465430298487

Hein, G., & Singer, T. (2008). I feel how you feel but not always: The empathic brain and its modulation. *Current Opinion in Neurobiology, 18*, 153-158. doi:10.1016/j.conb.2008.07.012

Hevner, A. R. (2007). A Three Cycle View of Design Science Research. *Scandinavian Journal of Information System, 19* (2, Article 4).

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design Science in Information Systems. *MIS Quarterly, 28*(1), 75–105.

Hoffman, M. L. (1982). Development of prosocial motivation: empathy and guilt. In N. Eisenberg, *The Development of Prosocial* (pp. 281-313). New York: Academic Press.

Hogan, R. (1969). Development of an empathy scale. *Journal of Consulting and Clinical, 33*, 307–316.

Hu, M., & Liu, B. (2004). Mining and summarizing customer reviews. *Proceedings of the 10th ACM SIGKDD international conference on knowledge discovery and data mining, KDD'04*, (pp. 168–177). New York.

Huang, Y., Lee, S., Ma, M., Chen, Y., Yu, Y., & Chen, Y. (2019). EmotionX-IDEA: Emotion BERT -- an Affectional Model for Conversation.

Hutto, C. J., & Gilbert, E. (2015). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. *Proceedings of the Eighth International AAAI Conference on Weblogs and Social Media.* Ann Arbor, MI.

Ickes, W. (1997). *Empathic Accuracy.* New York: The Guilford Press.

Janson, A., & Thiel de Gafenco, M. (2015). Engaging the appropriation of technology-mediated learning services - A theory-driven design approach. *23rd European Conference on Information Systems (ECIS'15).* Münster, Germany.

Johnson, R. H., & Blair, J. A. (1994). *Logical Self-Defense.* New York: McGraw-Hill Inc.

Jolliffe, D., & Farrington, D. P. (2006). Development and validation of the Basic Empathy Scale. *Journal of Adolescence, 29*, 589–611.

Kaitlin, L., & Konrath, S. H. (2019, December). Speaking of Psychology: The Decline of Empathy and the Rise of Narcissism. *Episode 95*. American Psychological Association.

Karl, K. A., O'Leary-Kelly, A. M., & Martocchio, J. J. (1993). The impact of feedback and self-efficacy on performance in training. *Journal of Organizational Behavio, 14*, 379–394. doi:10.1002/job.4030140409

Khanpour, H., Caragea, C., & Biyani, P. (2017). Identifying Empathetic Messages in Online Health Communities. *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)* (pp. 246–251). Taipei, Taiwan: Asian Federation of Natural Language Processing.

Konrath, S. H., O'Brien, E., & Hsing, C. (2011). Changes in Dispositional Empathy in American College Students Over Time: A Meta-Analysis. *Personality and Social Psychology Review, 15*(2), pp. 180-198. doi:10.1177/1088868310377395

Krippendorff, K. (1980). *Content Analysis: An Introduction to Methodology.* Beverly Hills, CA: Sage Publications, Inc.

Krippendorff, K. (2004). Measuring the reliability of qualitative text analysis data. *Quality and Quantity, 6*, pp. 787–800.

Kuechler, B., & Vaishnavi, V. (2008). On theory development in design science. *European Journal of Information Systems, 17*(5), 489-504. doi:10.1057/ejis.2008.40

Lawrence, J., Shaw, P., Baker, D., Baron-Cohen, S., & David, A. S. (2004). Measuring empathy: Reliabilityand validity of the Empathy Quotient. *Psychological Medicine, 43*(5), 911–919. doi:10.1017/S0033291703001624

Lehmann, K., Söllner, M., & Leimeister, J. M. (2015). Der Wert von ITgestütztem Peer Assessment zur Unterstützung des Lernens in einer Universitären. *Wirtschaftsinformatik (WI) Konferenz 2015.* Osnabrücke, Germany.

Li, Q., Hongshen, C., Zhaochun, R., Chen, Z., Tu, Z., & Ma, J. (2019). EmpGAN: Multi-resolution Interactive Empathetic Dialogue Generation.

Li, Y., Su, H., Shen, X., Li, W., Cao, Z., & Niu, S. (2017). DailyDialog: A Manually Labelled Multi-turn Dialogue Dataset. *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (pp. 986–995). Taipei, Taiwan: Asian Federation of Natural Language Processing.

Liddy, E. (2001). Natural Language Processing. In M. Decker, *Encyclopedia of Library and Information Science* (Vol. 2). NY: Inc.

Lippi, M., & Torroni, P. (2016). MARGOT: A web server for argumentation mining. *Expert Systems with Applications, 65*(15), pp. 292-302. doi:10.1016/j.eswa.2016.08.050

Lishner, D. A., Batson, C. D., & Huss, E. (2011). Tenderness and sympathy: Distinct empathic emotions elicited by different forms of need. *Personality and Social Psychology Bulletin, 37*, 614–625. doi:10.1177/0146167211403157

Liu, N., & Carless, D. (2006). Peer feedback: The learning element of peer assessment. *Teaching in Higher Education, 11*(3), 279-290. doi:10.1080/13562510600680582

Lopez-Perez, V. M., Perez-Lopez, C. M., & Rodriguez-Ariza, L. (2011). Blended learning in higher education: Students' perceptions and their relation to outcomes. *Computers and Education, 56*(3), 818-826. doi:10.1016/j.compedu.2010.10.023

Makarenkov, V., Rokach, L., & Shapira, B. (2019). Choosing the right word: Using bidirectional LSTM tagger for writing support systems. *Engineering Applications of Artificial Intelligence, 84*, 1–10.

March, S. T., & Smith, G. F. (1995). Design and Natural Science Research on Information Technology. *Decision Support Systems, 15*(4), 251–266. doi:10.1016/0167-9236(94)00041-2

Mehrabian, A. (1996). *Manual for the Balanced Emotional Emotional Empathy Scale (BEES).* Monterey, CA: Albert Mehrabian.

Mehrabian, A., & Epstein, N. (1972). A measure of emotional empathy. *Journal of Personality, 40*(4), 525–543. doi:10.1111/j.1467-6494.1972.tb00078.x

Meyer, C. (2014). *A Brief Tutorial on Inter-Rater Agreement.* Darmstadt, Germany: Technische Universität Darmstadt.

Mikolov, T., Corrado, G. S., Chen, K., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. *Proceedings of the International Conference on Learning Representations (ICLR 2013).* Scottsdale, AZ, USA.

Mohammad , S., Bravo-Marquez, F., Salameh, M., & Kiritchenko , S. (2018). SemEval-2018 Task 1: affect in tweets. *Proceedings of The 12th International Workshop on Semantic Evaluation* (pp. 1–17). Association for Computational Linguistics.

Mohammed, S., Zhu, X., & Kiritchenko, S. (2014). Sentiment Analysis of Short Informal Text. *Journal of Artificial Intelligence Research, 50*. doi:10.1613/jair.4272

Moyers, T. B., Manuel, M. J., Miller, W. R., & Ernst, D. (2007). *Revised Global Scales: Motivational Interviewing Treatment Integrity 3.0 (MITI 3.0).* University of New Mexico: Center on Alcoholism, Substance Abuse and Addictions (CASAA).

Neumann, D. L., Chan, R. C., Boyle, G. J., Wang, Y., & Westbury, H. (2015). Measures of Empathy: Self-Report, Behavioral,. In G. Matthews, D. H. Saklofske, & G. J. Boyle,, *Measures of personality and social psychological constructs* (pp. 257–289). Elsevier Academic Press. doi: 10.1016/B978-0-12-386915-

Nielsen, F. (2011). A new ANEW: evaluation of a word list for sentiment analysis in microblogs. *Proceedings of the ESWC2011 workshop on 'Making Sense of Microposts': big things come in small packages*, (pp. 93-98). Heraklion, Crete.

Oliveira-Silva, P., & Gonçalves, O. F. (2011). Responding empathically: A question of heart, not a question of skin. *Applied Psychophysiology and Biofeedback, 36*, 201–207. doi:10.1007/s10484-011-9161-2

Pavett, C. M. (1983). Evaluation of the Impact of Feedback on Performance and Motivation. *Human Relations, 36*, 641-654. doi:10.1177/001872678303600704

Pavey, L., Greitemeyer, T., & Sparks, P. (2012). "I help because I want to, not because you tell me to": Empathy increases autonomously motivated helping. *Personality and Social Psychology Bulletin, 38*(5), 681–689. doi:10.1177/0146167211435940

Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global Vectors for Word Representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1532–1543). Doha, Qatar: Association for Computational Linguistics. doi:10.3115/v1/D14-1162

Pérez-Rosa, V., Mihalcea, R., Resnicow, K., Singh, S., & An, L. (2017). Understanding and Predicting Empathic Behavior in Counseling Therapy. *Proceedings of the 55th Annual Meeting of the*

Association for Computational Linguistics (Volume 1: Long Papers) (pp. 1426–1435). Vancouver, Canada: Association for Computational Linguistics.

Preston, S. D., & De Waal, F. B. (2002). Empathy: Its ultimate and proximate bases. *Behavioral and Brain Sciences,, 25*, 1–20. doi:10.1017/S0140525X02350015

Pustejovsky, J., & Stubbs, A. (2012). *Natural Language Annotation for Machine Learning.* Sebastopol, CA: O'Reilly Media, Inc.

Ragheb, W., Azé, J., Bringay, S., & Servajean, M. (2019). LIRMM-advanse at SemEval-2019 task 3: attentive conversation modeling for emotion detection and classification. *Proceedings of the 13th International Workshop on Semantic Evaluation* (pp. 251–255). Minneapolis, Minnesota, USA: Association for Computational Linguistics.

Ramaprasad, A. (1983). On the definition of feedback. *Behavioral Science, 28*(1), 4–13. doi:10.1002/bs.3830280103

Reid, C., Davis, D., Horlin, C., Anderson, M., Baughman, N., & Campbell, C. (2012). The kids' empathic development scale (KEDS): A multidimensional measure of empathy in primary school-aged children. *British Journal of Development, 31*, 231–256.

Reniers, R., Corcoran, R., Drake, R. J., Völlm , B., & Shryane, N. (2011). The QCAE: a Questionnaire of Cognitive and Affective Empathy. *Journal of Personality Assessment, 93*(1), 84–95. doi:10.1080/00223891.2010.528484

Rietsche, R., & Söllner, M. (2019). Insights into Using IT-Based Peer Feedback to Practice the Students Providing Feedback Skill. *Proceedings of the 52nd Hawaii International Conference on System Sciences.* Hawaii, USA.

Rietsche, R., Lehmann, K., Haas, P., & Söllner, M. (2017). The Twofold Value of IT-Based Peer Assess-ment in Management Information Systems Education. *13th International Conference on Wirt-schaftsinformatik (WI).* St. Gallen, Switzerland.

Roschelle, J. (2013). Special issue on CSCL: Discussion. *Educational Psychologist, 48*(1), 67–70. doi:10.1080/00461520.2012.749445

Rueckert, L., & Naybar, N. (2008). Gender differences in empathy: The role of the right hemisphere. *Brain and Cognition, 67*(2), pp. 162–167. doi:10.1016/j.bandc.2008.01.002

Sadler, D. R. (1989). Formative assessment and the design of instructional systems. *Instructional Science, 18*(2), 119–144.

Santos, B. S., Colaqo Junior, M., & Gois de Souza, J. (2018). An Experimental Evaluation of the NeuroMessenger: A Collaborative Tool to Improve the Empathy of Text Interactions. *2018 IEEE Symposium on Computers and Communications (ISCC)*, (pp. 573–579). Natal. doi:10.1109/ISCC.2018.8538442

Sarkar, D., Bali, R., & Sharma, T. (2018). *Practical Machine Learning with Python : A Problem-Solver's Guide to Building Real-World Intelligent Systems.* La Fuente: California Apress.

Scherer, K. R., & Wallbott, H. G. (1994). Evidence for universality and cultural variation of differential emotion. *J Personal Soc Psychol, 66*(2), 310–328.

Sedoc, J., Buechel, S., Nachmany, Y., Buffone, A., & Ungar, L. (2020). Learning Word Ratings for Empathy and Distress from Document-Level User Responses. *Proceedings of The 12th Language Resources and Evaluation Conference (LREC 2020)* (pp. 1657–1666). Marseille: European Language Resources Association (ELRA).

Singer, T., & Lamm, C. (2009). The social neuroscience of empathy. *Annals of the New York Academy of Sciences,, 1156*, 81–96. doi:10.1111/j.1749-6632.2009.04418.x

Spreng, N. R., McKinnon, M. C., Mar, R. A., & Levine, B. (2009, January). The Toronto Empathy Questionnaire. *Journal of Personality Assessment, 91*(1), 62–71.

Stab, C., & Gurevych, I. (2014). Annotating Argument Components and Relations in Persuasive Essays. *Proceedings of the the 25th International Conference on Computational Linguistics (COLING 2014)*, (pp. 1501–1510). Dublin, Irland.

Strapparava, C., & Mihalcea, R. (2007). Semeval-2007 task 14: affective text. *Proceedings of the 4th international workshop on semantic evaluations, SemEval'07* (pp. 70–74). Stroudsburg: Association for Computational Linguistics.

Strappavara, C., & Valitutti, A. (2004). Wordnet-affect: an affective extension of wordnet. *Proceedings oft he 4th international conference on language resources and evaluation (LREC-2004)* (pp. 1083–1086). Lisbon: European Language Resources Association (ELRA).

Swamynathan, M. (2017). *Mastering machine learning with Python in six steps: a practical implementation guide to predictive data analytics using Pytho.* La Fuente: Apress.

Titchener, E. (1909). *Experimental psychology of the thought process.* New York: Macmillan.

Toulmin, S., Rieke, R., & Janik, A. (1984). *An Introduction to Reasoning.* New York: Macmillan Publishing Co. Inc.

Toxtli, C., Monroy-Hernanandez, A., & Cranshaw, J. (2018). Understanding Chatbot-mediated Task Management. *CHI '18: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 1–6. doi: 10.1145/3173574.3173632

UNESCO. (2017). *Education for Sustainable Development Goals.* Paris: UNESCO.

Vasilev, I., Slater, D., Spacagna, G., Roelants, P., & Zocca, V. (2019). *Python Deep Learning : Exploring Deep Learning Techniques and Neural Network Architectures with Pytorch, Keras, and TensorFlow* (2nd Edition ed.). Birmingham Packt Publishing Ltd.

Vaswani, A., Shazeer, N., Parmar, N., Uszokreit, J., Jones, L., Gomez, A., . . . Polosukhin, I. (2017). Attention Is All You Need. *Advances in Neural Information Processing Systems,*, (pp. 6000–6010). Long Beach.

Venkatesh, V., & Bala, H. (2008). Technology acceptance model 3 and a research agenda on interventions. *Decision Sciences, 39*(2), pp. 273–315.

Vollmeyer, R., & Rheinberg, F. (2005). A suprising effect of feedback on learning. *Learning and and Instruction, 15*, 58–-602. doi:10.1016/j.learninstruc.2005.08.001

Walton, D. (2009). Argumentation Theory: A Very Short Introduction. In I. Rahwan, & G. R. Simari, *Argumentation in Artificial Intelligence* (pp. 1-22). Boston: Springer-Verlag US.

Wambsganss, T., Leimeister, J.-M., Ruckstuhl, C., Handschuh, S., & Niklaus, C. (2020a). A Corpus for Modelling Empathy in Student-Written Peer Reviews. *Workingpaper*.

Wambsganss, T., Niklaus, C., Cetto, M., Söllner, M., Handschuh, S., & Leimeister, J.-M. (2020b). AL: An Adaptive Learning Support System for Argumentation Skills. *CHI '20: Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (pp. 1–14). Honolulu, HI, USA: CHI. doi:10.1145/3313831.3376732

Wang, Y., Feng, S., Wang, D., Yu, G., & Zhang, Y. (2016). Multi-label chinese microblog emotion classification via convolutional neural network. In F. Li, K. Shim, K. Zheng , & G. Liu, *Web technologies and applications: APWeb 2016. Lecture Notes in Computer Science* (Vol. 9931, pp. 567–580). Cham: Springer. doi:10.1007/978-3-319-45814-4_46

Webster, J., & Hackley, P. (1997). Teachning effectiveness in technology-mediated distance learning. *The Academy of Management Journal, 40*(6), 1282–1309. doi:10.2307/257034

Weiss, S. M., Indurkhya, N., Zhang, T., & Damerau,, F. (2005). *Text Mining. Predictive Methods for Analyzing Unstructured Information.* New York: Springer-Verlag NY.

Westbury, R. H., & Neumann, D. L. (2008). Empathy-related responses to moving film stimuli depicting human and non-human animal targets in negative circumstances. *Biological Psychology, 78*(1), 66–74. doi:10.1016/j.biopsycho.2007.12.009

Winkler, R., Büchi, C., & Söllner, M. (2019). Improving Problem-Solving Skills with Smart Personal Assistants: Insights from a Quasi Field Experiment. *ICIS Interantional Conference on Information Systems.* Munich, Germany: ACM Digital.

Wispé, L. (1987). History of the concept of empathy. In N. Eisenberg, & J. Strayer, *Empathy and its development* (pp. 17–39). Cambridge: Cambridge University Press.

Xiao, B., Can, D., Georgiou, G., Atkins, D., & Narayanan, S. (2012). Analyzing the language of therapist empathy in Motivational Interview based psychotherapy. *Proceedings of The 2012 Asia Pacific Signal and Information Processing Association Annual Summit and Conference,* (pp. 1–4). Hollywood, CA, USA: IEEE.

Xiao, B., Imel, Z., Georgiou, P., Atkins, D., & Narayanan, S. (2015). "Rate My Therapist": Automated Detection of Empathy in Drug and Alcohol Counseling via Speech and Language Processing. *PLoS ONE, 10*(12). doi:10.1371/journal.pone.0143055

Yin, D., Meng, T., & Chang, K. (2020). SentiBERT: A Transferable Transformer-Based Architecture for Compositional Sentiment Semantics. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (pp. 3695–3706). Online: Association for Computational Linguistics.

8 APPENDIX

A: Annotation Guidelines

Quick Guide

Annotation of empathy in student-written peer reviews

St. Gallen, May 2020

TABLE OF CONTENTS

List of Figures

1. Introduction

Nowadays, most information is readily available to people and solely reproduction of information is losing attention. This manifests in a shift of job profiles towards interdisciplinary, ambiguous and creative tasks (vom Brocke et al., 2018). Therefore, educational institutions need to evolve in their curricula, especially regarding the compositions of skills and knowledge conveyed. Particularly teaching higher order thinking skills to students, such as critical thinking, collaboration or problem-solving, has become more important during the last few years (Fadel et al., 2015). This has already been recognized by the Organization for Economic Co-operation and Development (OECD), which included these skills as a major element of their Learning Framework 2030 (OECD, 2018). One elementary skill for communication and successful team work represents the "*ability to simply understand the other person's perspective […] and to act emotionally on the other"* (Spencer 1870*),* also defined as empathy (Davis 1983).
However, studies have shown that empathy skills of students have decreased from 1979 to 2009 by more than thirty percent and even more rapid in the last period of the study from 2000 to 2009 (Konrath et al. 2011).
Therefore, our aim is to create an adaptive empathy learning tool, that supports student with a learning environment to improve being more empathetic. By creating a writing-support interface, students will get instant feedback from a pre-trained algorithm on their degree of empathy in written texts, e.g., when writing a peer-review on a fellow student's business idea. However, in order to leverage recent methods from Natural Language Processing and Machine Learning, a high-quality annotated corpus is needed to train a predictive model. This guidelines helpsto evaluate the corpus on empathy and foster a shared understanding on how empathy in student-written peer reviews can be detected.

2. Empathy in a nutshell

Despite the fact that everyone has a rough understanding of what it means to be empathetic, there is a broad variety of different definitions and operationalizations of empathy (Büchel, Buffone, Slaff, Ungar & Sedoc, 2018). Titchener (1909) first connected the German term "*Einfühlung*" to the English term empathy. However, empathy cannot just be described by one word, but rather consists of many different components. For the sake of these guidelines and in order to be able to annotate a German corpus of peer reviews, the following general understanding of empathy is sufficient: "***ability to react to the observed experiences of another […] and simply understand the other person's perspective*** "(Davis 1983, p. 1). Furthermore, empathy can be divided into various categories and subscales. Davis' (1983) proposed in his studies the four scales *fantasy scale* (imaginatively transpose oneself into fictional situations), *perspective taking* (ability to shift perspectives), *empathic concern* (degree to which the respondent experiences feelings of warmth, compassion and concern for the observed individual) and *personal distress* (individual's own feelings of fear, apprehension and discomfort at witnessing the negative experiences of others). Other authors and today's widely accepted distinction of empathy distinguishes between emotional (affective) and cognitive empathy, whereas emotional empathy lets us feel what others are feeling and cognitive empathy is the human's ability to recognize and understand others (Decety & Jackson, 2004; Lawrence, Shaw, Baker, Baron-Cohen & Davids, 2004; Jolliffe & Farrington, 2006; Gini, Albiero, Benelli & Altoe, 2007).

3. Data domain: student-written peer review

This annotation study is conducted on a set of German student-written peer reviews. The data was collected throughout a mandatory course of the master's program in Business Innovation at the University of St. Gallen. In this course, the students were asked to develop and present a new business model. Each student then received three different peer reviews, in which a fellow student from the same course elaborated on the strengths and weaknesses of the business model and gave persuasive recommendations and suggestions for improvement. The reviews were submitted online through a learning platform. The dataset for this annotation study contains a random subset of 500 peer reviews, collected from more than 7,000 documents over the last few years.

4. Empathy in student-written peer reviews

This chapter gives concrete guidelines on how to define empathy in the given peer review dataset and aims to establish a shared understanding of empathy in review texts. According to the elaboration of empathy mentioned in the first chapter, both approaches and scales are taken into consideration. However, since the reviews are evaluated on activities based on a new business idea of the student, Davis' fantasy scale and personal distress do not match. The fantasy scale denotes the tendency to transpose oneself into fictional characters in books, movies or plays. Since the data domain is about real-life business models, it does not represent such an environment of fictional characters. Additionally, students did not express personal negative experiences in their business models but rather present their business model in a logical, factual, and convincing manner. Thus, the scale of personal distress will not be included either. This leaves us with Davis' perspective taking and empathetic concern, as well as with cognitive and emotional empathy. Both approaches can be put together and applied to the context of peer reviews according the following:

- *Cognitive empathy (perspective taking):* The students use cognitive processes such as role taking, perspective taking or "decentering"[24] while evaluating the peers' submitted tasks. This means students set aside their own perspective and "step into the shoes of the other". Cognitive empathy can happen purely cognitive in that there is no reference to any affective state (Baron-Cohen & Wheelwright, 2004), but mostly includes understanding the other's emotional state as well. The following extract from a student-written peer review, translated to English, demonstrates high cognitive empathy: *"You could then say, for example, 'Since market services are not differentiated according to customer segments and locations, the following business areas result... And that due to the given scope of this ITPA you will focus on the Concierge-Service business segment.' After that, you have correctly only dealt with this business segment."* When annotating, it is helpful to read the statements from the feedback-receiver's perspective and think about if the reviewer is trying to help you with your further elaboration of your business idea and if he/she truly tries to put himself/herself in your shoes and find important hints, thoughts or comments.
- *Emotional empathy (empathic concern):* The students respond emotionally to the peers' affective state. The students can either show the same emotions as read in the review or simply state an appropriate feeling towards the peer. Typical examples include sharing excitement with the peer about the business model submitted or showing concern towards the peer's opinion The following extract from a student-written peer review, translated to English, shows high emotional empathy: *"I think your idea is brilliant!"* When annotating, it is helpful to read the statement from the feedback-writer's perspective and think about if he/she managed to respond emotionally to the peer's business idea by showing excitement, concern, disbelief, etc.

Each element of empathy will be evaluated on a scale from 1-5 (see Figure 1).

[24] Responding nonegocentrally (Piaget, 1932).

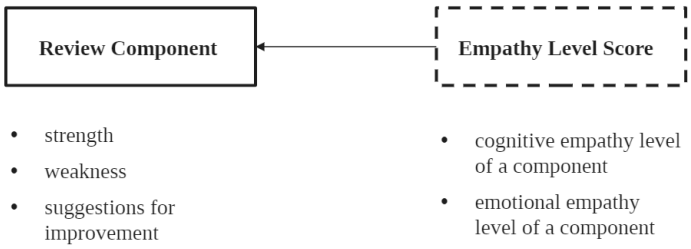


Figure 1: Annotation model

The differences between cognitive and emotional empathy should always be kept in mind. Statements can both be similar or very different in regards of emotional and cognitive empathy. The following example shows a component with high cognitive, but low emotional score: *Vielleicht noch ein wenig mehr ins Detail gehen und sich überlegen, wie die einzelnen Funktionen aussehen. Am Anfang ein paar weniger Funktionen anbieten und genauer auf einen Kundenwunsch eingehen. Ist das Kernprodukt, dass ich etwas mit meiner Freundin zusammen planen kann, oder der Aspekt es danach zu Veröffentlichen oder ist es einfach nur ein Tool, mit dem ich einfacher meine Daten zu meinem Urlaub sortieren kann? Ausgangs Pain ist ja die Überforderung mit zu viel Information. Wie genau behebst du das? Du fügst ja eher noch eine Informationsebene dazu indem man sich zusätzlich noch Boards von Freunden anschauen kann..*
The student did manage to put himself/herself into the peer's perspective. However, he/she did hardly show personal emotions.
The same concept can be applied for the other case. The following example illustrates a high emotional, but low cognitive score: *Ich finde deine Idee wirklich sehr, sehr gut!*
The student did manage to show a lot of excitement towards the peer's business idea. However, since it is missing any further explanations or supporting sentences (e. g. what particular is good about the idea), it receives a low cognitive score.

The following tables includes more details about the cognitive and emotional empathy scores. Because the general assessment of cognitive empathy of strengths, weaknesses and suggestions for improvement varies between the components, each component is defined specifically. This is not necessary for the assessment of emotional (affective) empathy due to the possibility to generalize the evaluation criteria.

Cognitive empathy	
1 = absolutely weak	The student's review is very short and does not include the peer's perspective. *Strengths*: The student only mentions one strength. This might not be relevant at all and lacks any further explanation, detail or example. *Weakness:* The student only mentions one weakness. This might not be relevant at all and lacks any further explanation, detail or example. *Suggestions for improvement:* The student only mentions one suggestion. The suggestion is not followed by any explanation or example and might not be relevant for the further revision of the peer.
2 = very weak	The student did not try to understand the peer's perspective. The student rather just tried to accomplish the task of giving feedback. *Strengths:* The student mentions one or more strengths. They could be relevant for the peer. However, he does not add any further explanation or details. *Weaknesses:* The student states one or more weaknesses without explaining why they are seen as such. They could be relevant for the peer. However, the statements do not include any further elaboration on the mentioned weakness. *Suggestions for improvement:* The student suggests one or more improvements that could be relevant for the peer. However, the student does not explain why he/she suggests the change or how the suggestions for improvement could be implemented.
3 = slightly weak / equal	The student tries to understand the perspective of the peer and adds further elaborations on his statements. However, his elaborations are not completely thought-through and his feedback is missing some essential explanations, examples, or questions to make sure he/she understood right. *Strengths:* The student mentions one or more strengths and explains some of them with minor explanations or examples on why it is seen as a strength. However, most strengths focus on formal aspects rather than contextual aspects. *Weaknesses:* The student states one or more weaknesses and explains some of them with minor explanations or examples. The student could also just state questions to illustrate the weakness in the peer's business idea. Most weaknesses are not explained why they are such. *Suggestions from improvements:* The student suggests one or more improvements that are mostly relevant for the further establishment of the activity. The suggestions are written only on a high-level and most of them do not include further explanations or examples. The student explains only occasionally *why* he/she suggests a change or how it could be implemented.
4 = Fairly strong	The student thinks from the perspective of the peer. He/She elaborates in a way that serves best the peer to further establish the idea or activity. Each component is affirmed with further explanations. *Strengths:* The student was able to recognize one or more strengths that are helpful for the peer to affirm their business idea and activity. He/She highlights contextual strengths rather than formal strengths. The student supports most statements with examples or further personal thoughts on the topic but might still be missing some reasonings. *Weaknesses:* The student thinks from the peer's perspective and what would help him/her to further succeed with the task. This could be demonstrated by stating various questions and establishing further thoughts. The student explains the weakness and adds examples, but he/she is still missing some reasonings. *Suggestions for improvement:* The student suggests one or more improvements that are relevant for the further establishment of the activity and idea from the perspective of the peer. Most suggestions are written concrete and, if applicable, supported by examples. In most cases, the student explains *why* he/she suggests a change.

5 = strong	The student fully understands the peer's thoughts. He/She completely stepped outside his/her own perspective and thinks from the peer's perspective. He/she does that by carefully evaluating the peer's idea according to its strengths, weaknesses and suggestions for improvement. Questions, personal pronouns or direct addressing of the author could be used in order to better understand and elaborate on the peer's perspective. *Strengths:* The student fully grasps the idea of the peer. He/She elaborates on strengths that are important for the peer for his continuation of the task and adds explanations, thoughts or examples to his statements, reasoning why the strength is important for the business idea. *Weaknesses:* The student thinks completely from the peer's perspective and what would help him/her to further succeed with the task. The student explains the weakness in a very detailed manner and describes why the weakness is important to consider. He could also give counter-arguments or ask questions to illustrate the weakness. *Suggestions for improvement:* The student suggests improvements as if he would be in the peer's perspective in creating the best possible solution. The student completes his suggestions with rich explanations on *why* he/she would do so and elaborates on the improvements in a very concrete and detailed way. Almost every suggestion is supported by further explanations.

Examples (Suggestions for improvement)

1. Auf der 2ten Slide in der Legende zwischen Gast und Restaurant. fehlt ein d, das muss hinzugefügt werden.
    - *Only one statement*
    - *Not relevant for the further elaboration of the peer's business idea and tasks*
    - *Written objectively and without personal pronouns*
2. Ich würde noch überlegen, ob du die Kunden besser einbeziehen kannst. Zudem musst du das BPMN nochmals überarbeiten.
    - *Two statements, they could be relevant for the peer*
    - *No further explanations or examples, why the peer should consider doing these changes*
3. Konkret würde ich am Anfang zwei Aufgaben nehmen Registrierung, Kreditanfrage und von Letzterer aus eine Nachricht an die Plattform senden. Da erscheint ein NachrichtenSymbol das in zwei aufeinanderfolgenden automatischen Aufgaben erst Vorprüfung, dann Vorauswahl mündet. Anschliessend ein exklusives Gateway mit zwei Pfeilen positiv, negativ. Negativ Aufgabe Absage versenden über eine EndeNachricht dunkler Kreis wieder zum Kreditnehmer in die Aufgabe Bescheid erhalten. Positiv Pfeil zu automatischer Aufgabe mit Aufforderung Daten zur Verfügung stellen und Nachricht in die Aufgabe beim Kreditnehmer Bescheid erhalten. Beim Kreditnehmer Swimlane würde ich ein Gateway mit zwei Pfeilen Anfrage abgelehnt, Anfrage genehmigt. Bei Ablehnung erfolgt das EndeSymbol. Bei einer Genehmigung folgt die Aufgabe Zusätzliche Daten auf Plattform laden. Danach erfolgt beim Kreditnehmer nur noch die Aufgabe Bescheid erhalten. Bei der Plattform folgt die Aufgabe Konsolidierung der Daten Durch Sequenzfluss folgend auf die Nachricht nach der Aufgabe Aufforderung Daten zur Verfügung stellen. Danach die beiden manuellen Aufgaben aus der Aufgabenstellung und die Aufgabe Freischaltung auf der Plattform. Nun stosst die Swimlane des Anlegers dazu. Ich würde eine Schleifenaufgabe bei Registrierung der Zusagen einbauen.
    - *More than two statements that are mostly relevant*
    - *Some suggestions are supported by further explanation or example*
    - *However, most statements are missing further elaboration on why the suggestions is made*

4. Vielleicht noch ein wenig mehr ins Detail gehen und sich überlegen, wie die einzelnen Funktionen aussehen. Am Anfang ein paar weniger Funktionen anbieten und genauer auf einen Kundenwunsch eingehen. Ist das Kernprodukt, dass ich etwas mit meiner Freundin zusammen planen kann, oder der Aspekt es danach zu Veröffentlichen oder ist es einfach nur ein Tool, mit dem ich einfacher meine Daten zu meinem Urlaub sortieren kann? Ausgangs Pain ist ja die Überforderung mit zu viel Information. Wie genau behebst du das? Du fügst ja eher noch eine Informationsebene dazu indem man sich zusätzlich noch Boards von Freunden anschauen kann...
    - *Several statements that are relevant for the further elaboration on the peer's tasks*
    - *Most statements are supported by further elaborations (e. g. specific questions to trigger more thoughts on the topic)*
    - *Most statements are explained why the are suggested (e. g. by showing a fact that has been missed)*
    - *Some elaborations could be written more concrete*
5. Ein nächster Schritt wäre eine Analyse der tatsächlichen Kaufkraft der Kunden und Anzahl potentieller Benutzer zu machen. Vor allem ist hierbei wichtig herauszufinden wie viele Bestellungen/Benutzer benötigt sind um Profit zu machen. Um das Netzwerk weiter auszubauen wird empfohlen mit Partnerorganisationen, wie Sportvereine, Vegane/Vegetarische Hersteller, und weitere zusammenzuarbeiten. Dadurch entsteht eine breite Produktpalette, die kundenspezifisch zugeschnitten werden kann. Damit eine erfolgreiche Zusammenarbeit zwischen Hersteller und Kunden generiert wird, wird empfohlen nochmals tiefgründig über ihre Beziehung zu gehen und weitere Möglichkeiten der Zusammenarbeit aufzuzeigen. Die Kundendaten allein generieren bereits Wert für den Hersteller, jedoch würde eine Art Innovation Lab, mit Meetings zwischen Herstellern und Kunden die Beziehung beispielsweise stärker gestalten. Hersteller können hierbei mit Hilfe von Kunden neue Produkte entwickeln, und diese dann von Kunden direkt testen lassen. Um die Abonnement Gebühr für Kunden so tief wie möglich zu gestalten, könnten Rabattaktionen erarbeitet oder mit Gutscheinen geworben werden. Zudem könnte man für jede Weiterempfehlung an neue Kunden eine Box gratis zur Verfügung stellen. In der Lösung besteht viel Potential was weiter erarbeitet werden müsse. Nebst der ZielgruppenAnalyse wird empfohlen, sich tiefgründig mit dem Produktangebot, sowie Pricing beschäftigen. Zudem wird empfohlen, die Beziehung zu Herstellern und zwischen Herstellern genauer zu definieren und Lösungen aufzuzeigen, um Konkurrenz innerhalb der Gruppen zu vermeiden.
    - *Several statements that are all relevant for the peer's idea*
    - *Statements are supported by rich explanations and further details*
    - *Suggestions are explained why they are suggested and why they should be considered*

Emotional (affective) empathy	
1 = absolutely weak	The student does not respond emotionally to the peer's work at all. He/She does not show his/her feelings towards the peer and writes objectively (e.g. no "I feel", "personally" "I find this.." and no emotions such as "good", "great", "fantastic", "concerned", etc.). Typical examples would be "add a picture." or "the value gap XY is missing."
2 = very weak	Mostly, the student does not respond emotionally to the peer's work. Only very minor and weak emotions or personal emotional statements are integrated. The student writes mostly objectively (e. g. "okay", "this should be added", "the task was done correctly", etc.). In comparison to 1, he/she might be using modal verbs (might, could, etc.) or words to show insecurity in her review (rather, maybe, possibly)
3 = slightly	The student *occasionally* includes emotions or personal emotional statements to the peer re-

	view. They could be quite strong. However, the student's review is missing personal pronouns ("I", "You") and is mostly written in third person. Emotions can both be positive or negative. Negative emotions can be demonstrated with concern, missing understanding or insecurity (e. g. with modal verbs or words such as rather, perhaps). Typically, scale 3 includes phrases such as "it's important", "the idea is very good", "the idea is comprehensible", "it would make sense", "the task was done very nicely", "It could probably be, that", etc.
4 = Fairly strong	The student was able to respond emotionally to the peer's submitted activity with suitable emotions (positive or negative). He/She returns emotions in his/her review on *various* locations and expresses his/her feelings by using the personal pronoun ("I", "You"). Some sentences might include exclamations marks (!). Typical reviews in this category include phrases such as "I am excited", "this is very good!", "I am impressed by your idea", "I feel concerned about", "I find this very..", "In my opinion", "Unfortunately, I do not understand", "I am very challenged by your submission", "I am missing", "You did a very good job", etc.
5 = strong	The student was able to respond very emotionally to the peer's work and fully represents the affectional state in his/her *entire* review. He/She illustrates this by writing in a very emotional and personal manner and expressing his/her feelings (positive or negative) throughout the review. Strong expressions include exclamation marks (!). Typical reviews in this category include phrases such as "brilliant!", "fantastic", "excellent", "I am totally on the same page as you", "I am very convinced", "personally, I find this very important, too", "I am very unsure", "I find this critical", " I am very sure you feel", "This is compelling for me" etc.

Examples (Suggestions for improvement)

1. Die USP des Konzeptes besser herausarbeiten und zeigen inwiefern sich dieses Konzept von den bisherigen Vermittlungsbüros unterscheidet.
    - *The student uses no personal emotions*
    - *Very objectively and "dry"*
2. Der Autor sollte sich nochmals genau mit den einzelnen Punkten des BMN auseinandersetzen und sich überlegen, wie die Geschäftsidee von einem Unternehmen umgesetzt/implementiert werden könnte.
    - *Rather factually*
    - *The student uses modal verbs (e. g. "der Autor sollte.")*
3. Ich würde noch auf die Schreibweise achten, damit dein tolles Beispiel nicht untergeht. Beispielsweise wurde beim Punkt Marktleistungen das Wort eine doppelt genannt. Der Abschnitt Kurz Charakteristika Ihres Unternehmens sollte vielleicht noch einmal überarbeitet werden, da noch sehr viele grammatikalische Fehler bestehen und somit der Lesefluss gehindert wird.
    - *The student occasionally illustrates emotions by using modal verbs and certain emotional expressions ("toll", "vielleicht")*
    - *The student only occasionally includes personal pronouns ("Ich würde"), but writes mostly in third person ("Beispielsweise wurde", ""sollte vielleicht nochmal überarbeitet werden".*
4. Bieten alle Skigebiete genügend Empfang? Als regelmässiger Skifahrer musste ich schon einige Male erfahren, dass es viele Funklöcher gibt. Konkretisiere, falls möglich, wie man mit diesem Problem umgehen kann. Wäre es nicht spannend, mit den Skiausrüstern an den Talstationen z.B. Intersport eine Partnerschaft anzustreben? Intersport bietet die Hardware, On the Top bietet die Software. Für Intersport ein super Deal, da die ein solches Angebot wahrscheinlich auch in fünf Jahren noch nicht hinkriegen würden. Bei den Konkurrenten wären möglicherweise noch andere App-Anbieter zu beachten, die ähnliche Angebote auf den Markt gebracht haben. Du erwähnst das teure Bergsport-Angebot in der Schweiz, womit du absolut recht hast. Ich nehme jedoch an, dass auch On the Top sich noch ein Stück vom Kuchen sichern will. Frage Wie kann

On the Top Geld verdienen und es dem Kunden erlauben, günstiger Skifahren zu gehen? Oder strebt das Unternehmen eher einen Added Value an?

- *The student shows personal emotions on various locations ("musste ich schon einige Male erfahren" shows annoyance, "wäre spannend» shows excitement, «womit du absolut recht hast» shows agreement)*
- *The student writes subjectively ("musste ich erfahren", "ich nehme jedoch an")*

5. Du musst zwingend eine Tabelle erstellen, welche die verschiedenen Geschäftsfeldkombinationen darstellt und dich dann auf eine beschränken. Die nachfolgenden Kapitel richten sich danach explizit nach diesem Geschäftsfeld aus. Auf S.2122 im Skript von Österle siehst du, wie du genau vorgehen sollst. Das Kapitel 7 Qualitative Beschreibung sollte ebenfalls dem Beispiel von Österle folgen. Du hast hierbei einige Unterpunkte vergessen zu erwähnen. Auch wenn deiner Ansicht nach bspw. keine Lieferanten vorhanden sind, solltest du das meiner Meinung nach zum Verständnis doch auch erwähnen. Ich verstehe durchaus, dass das ConciergePersonal für den Erfolg deiner Geschäftsidee extrem wichtig ist. Dennoch vergisst du auch, dass die Bootsanbieter und die nachfrager ebenfalls von grosser Bedeutung sind. Befinden sich auf der Plattform keine Bootsanbieter so ergibt sich keinen Nutzen für Bootsnachfrager und umgekehrt, wodurch sich die Plattform niemals etablieren wird. Ich denke, dass du das ebenfalls in deiner Lösung einarbeiten solltest. Die Massnahmen empfinde ich als eher verwirrend und ergeben in Zusammenhang mit vorherigen Kapiteln nicht wirklich Sinn. Du solltest hierbei ein wenig genauer werden und darauf achten, dass die Massnahmen gerade Punkt 1 und 3 mit bereits Beschriebenem übereinstimmt.

- *The student illustrates emotions throughout the entire review and uses strong words to demonstrate his emotion (e. g. concern) ("du musst zwingend eine Tabelle erstellen" "du solltest", "eher verwirrend", "ergeben nicht wirklich Sinn", "niemals", etc.)*
- *The student is using a lot of personal expressions ("Ich verstehen durchaus", "Dennoch vergisst du auch", "Ich denke, dass du", "empfinde ich")*

The dataset will be annotated according to the above-mentioned empathy elements on each *component* of the student-written peer review. This means that the evaluation of empathy will be applied to the description of the strengths, weaknesses and suggestions for improvement and will not be applied on a word or sentence basis. The following chapter will provide more detail on how to annotate the peer reviews.

5. Annotation process

The previous sections briefly described the components that are aimed to annotate in this study. For annotating these components, the annotation process is split into three steps: 1) reading of the entire review 2) labeling of the components and elaborations and 3) classification of both empathy scales.

1. *Reading of the entire peer review:* The annotators are confronted with the student-written peer review and are asked to read the whole document. This helps to get a first impression of the review and to get an overview of the single components and structure of it.
2. *Labeling the components and elaborations:* After reading the entire student-written peer review, the annotator is asked to label the three different components (strengths, weaknesses and suggestions for improvement). Details on how to label them can be found in the next chapter. Every supporting sentence (such as explanation, example, etc.) will be annotated together with the according component. Figure 2 illustrates how the components are annotated.

3. *Classification of both empathy scales:* Each component is assessed on its level of cognitive and emotional empathy by giving a number between 1-5. Each category is carefully defined and delimited.

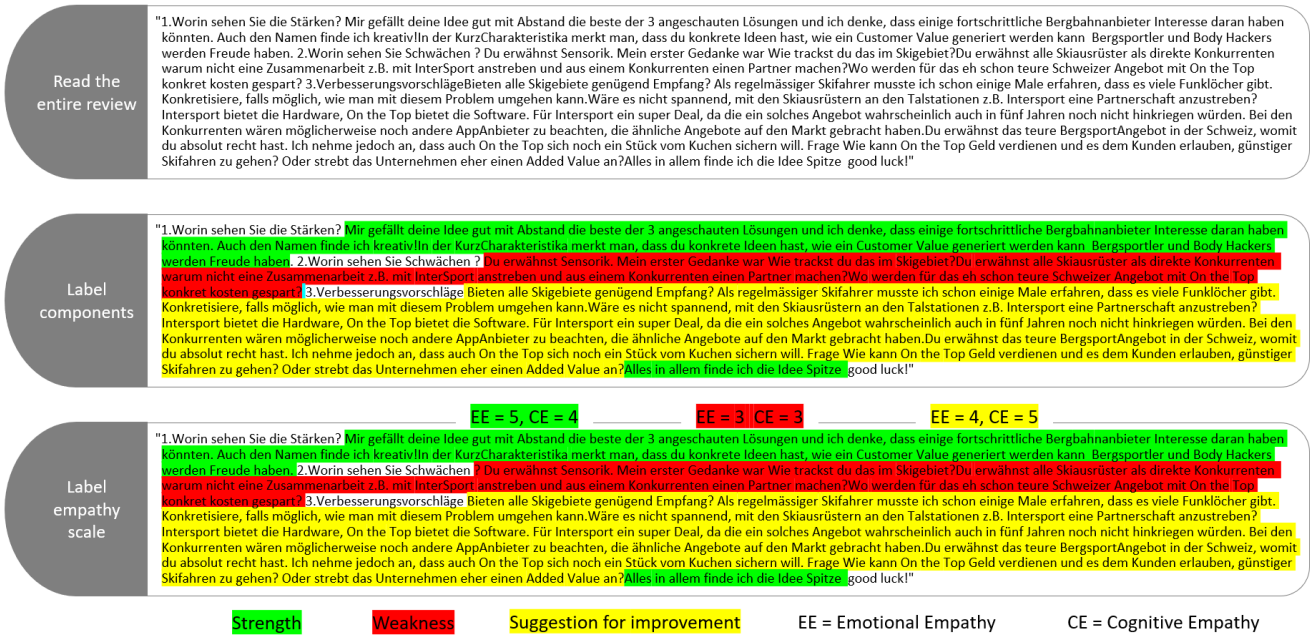


Figure 2: Annotation process

6. Boarders of review components

The structure of the student-written peer reviews can vary. Students were given the task to evaluate the peer's activity according to its strength, weaknesses and to give suggestions for improvement. They were not told on how to structure their reviews or on how long it should be. Therefore, significant differences can be detected in terms of structuring and length. Nevertheless, some guidelines can be derived to enhance annotation results. The following illustration helps to better understand them.

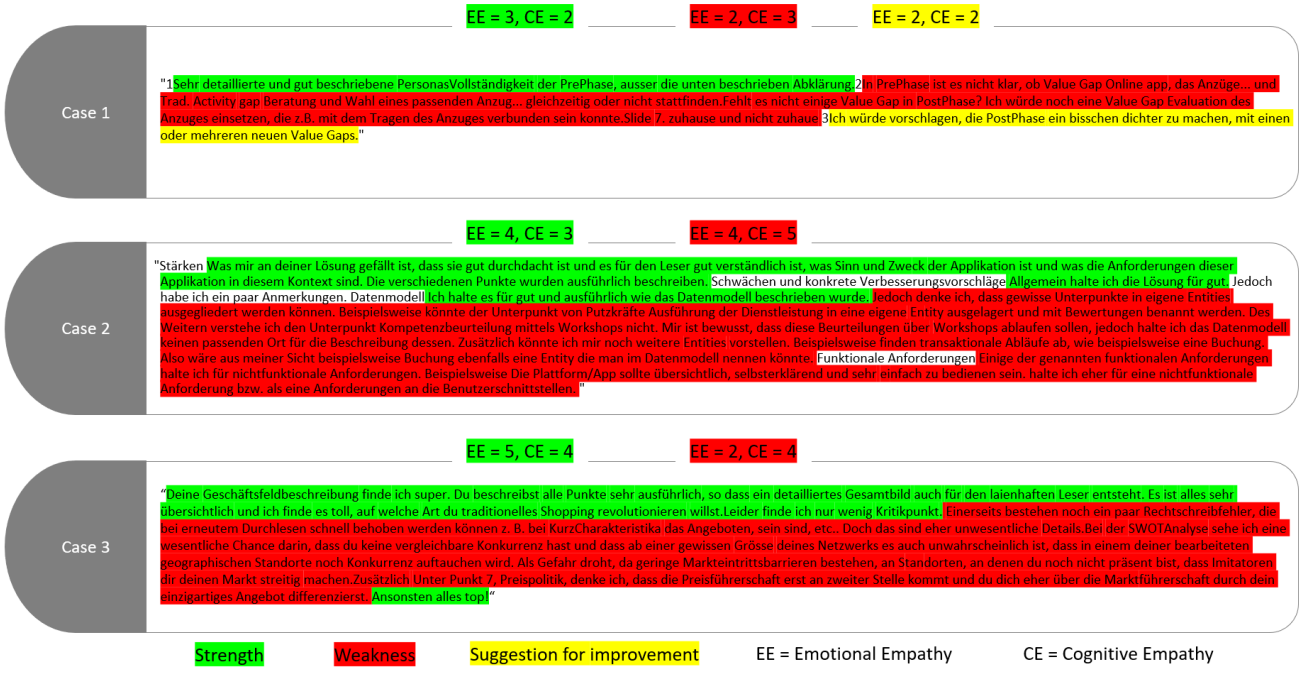


Figure 3: peer review structure

General

- Generally, the review will be annotated according to three components (1: Strength, 2: Weakness, 3: Suggestions for improvement). Thus, every sentence in the first chapter will be annotated as a strength, every sentence in the second chapter as a weakness and every sentence in the third chapter as a suggestion for improvement. See case 1 in Figure 3 as an example. However, there are two exceptions:
    - Sentences that clearly show a strength but are mentioned in the weakness or suggestions for improvement component. Those exceptions should be annotated according to their allocation. See case 3 in Figure 3 as an example ("Otherwise, all great!")
    - If weaknesses and suggestions for improvement are both combined in one component (e. g. Schwächen & Verbesserungsvorschläge), the whole component will be annotated as a weakness. See case 2 in Figure 3 as an example.
    - Some reviews might not follow a clear chapter structure. Those reviews are annotated to the best of the annotator's knowledge according to the rules applying to strength, weakness and suggestion for improvement. See case 2 in Figure 3 for an example, where the review is not structured according to the three chapters. The guidelines respecting the distinction between strength, weakness, suggestion can be found below. These rules only apply to reviews that are not following a clear structure.
- Greetings, names or farewell sentences (such as "Dear XX", "best wishes", "kind regards", etc.) will not be annotated. See Figure 2, where "good luck" is not annotated.
- Titles of the chapters (such as "Strength", "Suggestions for improvement"), numerations ("1.", "2.", "3."), but also other titles from the task (such as "Positive", "Key Resources", "Personas", "BPMN", etc.) will be ignored. See Figure 2, where "1. Worin sehen Sie Stärken?" or case 2 in Figure 3, where "Datenmodell" and "Funktionale Anforderungen" are not annotated
- Quotation marks at the beginning and at the end of the review will not be annotated. See Figure 3, where " at the beginning and " at the end are not annotated. Also, any other marks ("→" ">") will be ignored.
- Further information that do not belong to the review from a contextual point of view (such as "First, I'd like to let you know that I have previous knowledge in the field of your business idea", "This leads me to the following strengths" "I have the following suggestions for you") will not be annotated. See case 2 in Figure 3 where "Jedoch habe ich ein paar Anmerkungen" is not annotated.
- A sentence can consist of several components. The annotator is allowed to separate the sentence according to the component (e. g. "Ich finde deine Idee sehr gut, aber mir gefällt deine Darstellung nicht" is split into a strength component ("Ich finde deine Idee sehr gut") as well as a weakness component ("aber mir gefällt deine Darstellung nicht").
    - A component can consist of several sentences. These sentences can be directly followed by each other or can be separated trough other components.
    - Explanations, further elaborations, details, examples, etc. that support a component are annotated together with the component.
- Each component is assessed on its level of cognitive and emotional empathy
    - When annotating, it is important that the entire component (every sentence that has been marked as this component) will be evaluated. Components are not split up for the assessment of empathy. See Figure 2 and 3 as examples, where the sentences marked in green are all combined and together evaluated on their scale of emotional and cognitive empathy (and therefore only given one total score for each label per component).
    - If one of the components is missing (e.g. the peer review does not include any strengths), it will not be annotated and therefore not given any label regarding emotional and cognitive empathy.

Strengths

- Something positive about the peer's submitted work ("Your BPMN is very structured", "Your idea is very interesting", etc.)
- Something that the peer liked (e. g. "I like how you did")
- Can be general or very specific
- A "positive" weakness or suggestion for improvement (e.g. "I do not find any weakness")

Weakness[25]

- Something negative about the peer's submitted work, a point of criticism ("I do not see why", "I do not understand", "It does not make sense")
- Missing parts or thoughts in the peer's idea or task, something that the reviewee wish he would have done or further hypothetical considerations ("I would have liked that you", "I wish you would have", "I missed", "It would be interesting to see/to know", "one could integrate", I think it would be good to")
- Questions that are showing disagreement ("Wouldn't it be?", "Don't you think that?")
- No concrete instruction, order or action for the peer derived yet, *no personal form of address* ("you", "the author")

Suggestions for improvement

- Something that should be added for the second version of the peer's work ("For the second version", "You should add", etc.)
- Concrete suggestions or parts that should be improved or need more attention ("I suggest that *you*", "my suggestion is that *you*", "try to", I think it would be good that *you*")
- Concrete instructions, invitations, orders or actions, directed towards the peer ("*You* could", *You* should", "*The author* must", "Would it be possible that *you*", the use of direct instructions like "add a second box" or "integrate another sentence about", etc.)

7. References

Baron-Cohen, S., and Wheelwright, S. 2004. "The Empathy Quotient: An Investigation of Adults with Asperger Syndrome or High Functioning Autism, and Normal Sex Differences," *Journal of Autism and Developmental Disorders* (Vol. 34).

vom Brocke, J., Maaß, W., Buxmann, P., Maedche, A., Leimeister, J. M., and Pecht, G. 2018. "Future Work and Enterprise Systems," *Business and Information Systems Engineering* (60:4), pp. 357–366. (https://doi.org/10.1007/s12599-018-0544-2).

Davis, M. H. 1983. "Measuring Individual Differences in Empathy: Evidence for a Multidimensional Approach.," *Journal of Personality and Social Psychology* (44:1), pp. 113–126. (https://doi.org/10.1037//0022-3514.44.1.113).

Decety, J., and Jackson, P. L. 2004. "The Functional Architecture of Human Empathy.," *Behavioral and Cognitive Neuroscience Reviews*, pp. 71–100. (https://doi.org/10.1177/1534582304267187).

Fadel, C., Bialik, M., and Trilling, B. 2015. *Four-Dimensional Education : The Competencies Learners Need to Succeed*.

Konrath, S. H., O'Brien, E. H., and Hsing, C. 2011. "Changes in Dispositional Empathy in American College Students over Time: A Meta-Analysis," *Personality and Social Psychology Review* (15:2), pp. 180–198. (https://doi.org/10.1177/1088868310377395).

Lawrence, E. J., Shaw, P., Baker, D., Baron-Cohen, S., and David, A. S. 2004. "Measuring Empathy: Reliability and Validity of the Empathy Quotient," *Psychological Medicine* (34:5), pp. 911–919. (https://doi.org/10.1017/S0033291703001624).

OECD. 2018. *The Future of Education and Skills - Education 2030*. (https://doi.org/2018-06-15).

Peterson, R. T., and Limbu, Y. 2009. "The Convergence of Mirroring and Empathy: Communications Training in Business-to-Business Personal Selling Persuasion Efforts," *Journal of Business-to-Business Marketing* (16:3), pp. 193–219. (https://doi.org/10.1080/10517120802484551).

Spreng, R. N., McKinnon, M. C., Mar, R. A., and Levine, B. 2009. "The Toronto Empathy Questionnaire: Scale Development and Initial Validation of a Factor-Analytic Solution to Multiple Empathy Measures," *Journal of Personality Assessment* (91:1), Taylor and Francis Inc., pp. 62–71. (https://doi.org/10.1080/00223890802484381).

[25] A weakness often implies a suggestion, too. If is more of a point of criticism rather than a concrete instruction or suggestion directed to the peer, the sentence stays a weakness. A suggestion must be explicitly directed towards the peer ("you", "the author").

B: Source Codes

Data Preparation

```
#import modules
import sys, json
import nltk
nltk.download('punkt')
import pandas as pd
from pprint import pprint
import collections
import re
import glob
import seaborn as sns

#restart runtime before running
import spacy
from spacy.tokenizer import Tokenizer
nlp = spacy.load('de_core_news_sm')
tokenizer = Tokenizer(nlp.vocab)

"""Read JSON output from TagTog and create data table, saved as .ann.txt"""

def tokenize(text):
   doc = nlp(text)
   lst = [(token.idx, token.text) for token in doc]
   return [x for x in lst if len(x[1].strip()) > 0]

def sliding_window(lst, window_size):
   for i in range(0, len(lst)-window_size+1):
       yield lst[i:i+window_size]

def locate(s, text):
    def match(text_tokens, search_tokens):
        for i, (tt, st) in enumerate(zip(text_tokens, search_tokens)):
            if (i == 0) and tt[1].endswith(st[1]):
                pass
            elif (i == len(search_tokens)-1) and tt[1].startswith(st[1]):
                pass
            elif tt[1] == st[1]:
                pass
            else:
                return False
        return True

    text_tokens = tokenize(text)
    search_tokens = tokenize(s)

    for ngram in sliding_window(text_tokens, len(search_tokens)):
        if match(ngram, search_tokens):
            fr = ngram[0][0]
            to = ngram[-1][0] + len(ngram[-1][1])
            return fr, to
    return None

def read_json():
    j=0
    file_path = glob.glob("/Data/TagTog_output/txt/***.txt")
    for file_ in file_path:
      print(file_)
      txt=(re.split('/',file_)[-1])
```

```
        print(txt[:-4])
        file=open(file_)
        input_text = file.read()
        tokens_input_text = nltk.word_tokenize(input_text)


        json_file="/Data/TagTog_output/json/"+txt[:-4]+".ann.json"
        json_data = open("/Data/TagTog_output/json/"+txt[:-4]+".ann.json",
"r")

        annots = []
        data = json.load(json_data)
        json_data.close()

        for i in range(0, len(data['entities'])):
            if data['entities'][i]['classId'] == 'e_1':
                strength = []
                offsets = data['entities'][i]['offsets']
                start = offsets[0]['start']
                text = offsets[0]['text']
                print(text)
                test = locate(text, input_text)
                if test:
                    start_strength = test[0]
                    length_strength = test[1]-start_strength
                else:
                    start_strength = input_text.find(text)
                    length_strength = len(text)

                fields = data ["entities"][i]["fields"]
                try:
                    f_4 = fields["f_4"]["value"]
                except KeyError:
                    f_4 = "not found"
                    print(json_file)

                try:
                    f_5 = fields["f_5"]["value"]
                except KeyError:
                    f_5 = "not found"
                    print(json_file)

                strength.append("strength")
                strength.append(start_strength)
                strength.append(length_strength)
                strength.append(f_4)
                strength.append(f_5)
                strength.append(text)
                annots.append(strength)
            elif data['entities'][i]['classId'] == 'e_2':
                weakness = []
                offsets = data['entities'][i]['offsets']
                text = offsets[0]['text']
                test = locate(text, input_text)
                if test:
                    start_weakness = test[0]
                    length_weakness = test[1]-start_weakness
                else:
                    start_weakness = input_text.find(text)
                    length_weakness = len(text)

                fields = data ["entities"][i]["fields"]
```

```
                try:
                    f_4 = fields["f_4"]["value"]
                except KeyError:
                    f_4 = "not found"
                    print(json_file)

                try:
                    f_5 = fields["f_5"]["value"]
                except KeyError:
                    f_5 = "not found"
                    print(json_file)

                weakness.append("weakness")
                weakness.append(start_weakness)
                weakness.append(length_weakness)
                weakness.append(f_4)
                weakness.append(f_5)
                weakness.append(text)
                annots.append(weakness)

            elif data["entities"][i]['classId'] == 'e_3':
                suggestions = []
                offsets = data['entities'][i]['offsets']
                text = offsets[0]['text']
                test = locate(text, input_text)
                if test:
                    start_suggestions = test[0]
                    length_suggestions = test[1]-start_suggestions
                else:
                    start_suggestions = input_text.find(text)
                    length_suggestions = len(text)

                fields = data ["entities"][i]["fields"]
                try:
                    f_4 = fields["f_4"]["value"]
                except KeyError:
                    f_4 = "not found"
                    print(json_file)

                try:
                    f_5 = fields["f_5"]["value"]
                except KeyError:
                    f_5 = "not found"
                    print(json_file)

                suggestions.append("suggestions")
                suggestions.append(start_suggestions)
                suggestions.append(length_suggestions)
                suggestions.append(f_4)
                suggestions.append(f_5)
                suggestions.append(text)
                annots.append(suggestions)

        f_out = open("/Data/ann1/"+txt[:-4]+".ann.txt","w")
        j=j+1
        for a in annots:
            for entry in a:
                f_out.write(str(entry) + '\t')
            f_out.write('\n')
        print(annots)
read_json()
```

```
"""Create Pandas Dataframe and map with original text, convert to .csv"""

def map_with_text():
  line_no=0
  dfObj = pd.DataFrame(columns=['UniqueID','DocumentID','clas-
sID','start','length','f_4','f_5','text']) #create pandas dataframe
  j=0
#Loop over files
  file_path = glob.glob("/Data/TagTog_output/txt/***.txt")
  for file_ in file_path:
    print(file_)
    txt=(re.split('/',file_)[-1])

    file=open(file_)
    input_text = file.read()
    print('file  is ',j)
    tokens_input_text = nltk.word_tokenize(input_text)

    f_out = open("/Data/ann1/"+txt[:-4]+".ann.txt", "r")
    r=f_out.readlines()

    previous_end_index=0
    for line in r:
      text=(re.split('\t',line)[-2])
      #print(line)
      print(text)
      match = input_text.find(text)
      print('match is ',match)
      print
      if match!=-1 and len(text)>0:
        start_index=match
        end_index=match+len(str(text))
        print(previous_end_index,' ',start_index)
        if previous_end_index<start_index:
          print('not equal')
          dfObj = dfObj.append({'UniqueID': line_no,'DocumentID': j,'clas-
sID': "None" ,'start': previous_end_index , 'length':(start_index-previ-
ous_end_index) , 'f_4': "None" , 'f_5': "None" , 'text': input_text[previ-
ous_end_index:start_index]}, ignore_index=True) ## saving in a dataframes
          line_no+=1
        previous_end_index=end_index
        class_id=str(re.split('\t',line)[0])
        f4=(re.split('\t',line)[3])
        f5=(re.split('\t',line)[4])
        dfObj = dfObj.append({'UniqueID': line_no,'DocumentID': j,'clas-
sID': class_id ,'start': start_index , 'length': end_index-start_index ,
'f_4':f4 , 'f_5': f5 , 'text': str(text) }, ignore_index=True) #saving in a
dataframe

        line_no+=1
    j=j+1
  return dfObj

dataset_complete=map_with_text()
dataset_complete

dataset_complete.to_csv(r'/Data/dataset_complete.csv', index=False)
```

```
"""Data Analysis"""

#Mount drive before running
df = pd.read_csv("/Data/dataset_groupedempathylevel.csv")
df.drop(df[df['length']<=3].index, inplace = True)
columns_to_keep = ['text','classID', 'f_4', 'f_5']
df = df[columns_to_keep]
df

import matplotlib.pyplot as plt
sns.countplot(df.classID, color="g")
plt.xlabel('ClassID')
plt.title('Distribution of review components')

import matplotlib.pyplot as plt
sns.countplot(df.f_4, color="g" )
plt.xlabel('f_4')
plt.title('Distribution of emotional empathy')

import matplotlib.pyplot as plt
sns.countplot(df.f_5, color="g")
plt.xlabel('f_5')
plt.title('Distribution of cognitive empathy')
```

LSTM

```
Set-Up

#Only do once
!git clone https://github.com/facebookresearch/fastText.git
!cd fastText
!pip install fastText

#Importing modules
import warnings
warnings.filterwarnings("ignore")
import pandas as pd
import numpy as np
import nltk
from sklearn import metrics
nltk.download('stopwords') #Downloading stopwords
import os
import random
import re
import pickle
import tensorflow as tf
from datetime import datetime
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import Input, LSTM, Bidirectional, Dense,
TimeDistributed
from tensorflow.keras.layers import Embedding, Flatten
from tensorflow.keras.layers import MaxPooling1D, Dropout, Activation,
Conv1D
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing import sequence
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint
from tensorflow.keras.models import load_model
from sklearn.metrics import f1_score, accuracy_score, confusion_matrix
from sklearn.model_selection import train_test_split
from keras.utils import to_categorical

#Loading dataset
df = pd.read_csv("/Data/dataset_groupedempathylevel.csv")
df.drop(df[df['length']<=3].index, inplace = True)  #droping all rows that
are smaller/equal 3 in length
columns_to_keep = ['text','classID', 'f_4', 'f_5'] #dropping the rest
df = df[columns_to_keep]
df

"""Data Preprocessing"""

df['text'] = df['text'].str.replace(r"[\d\.]+", "").str.strip() #Removing
digits
df['text'] = df['text'].str.replace("[^\w\s]", "").str.lower() #Converting
to lower case
german_stop_words = nltk.corpus.stopwords.words('german') #List of german
stopwords
df['text'] = df['text'].apply(lambda x: ' '.join([item for item in
x.split() if item not in german_stop_words])) #Removing stop words

# Converting categorical labels to numerical values
df["fn_4"] = df["f_4"].astype('category').cat.codes
df["fn_5"] = df["f_5"].astype('category').cat.codes

df
```

```
#Initializing parameters
CURR_PATH = !pwd
PATH_DATA = CURR_PATH[0]
PATH_MODELS = PATH_DATA + "/Data/LSTM/saved models"
PATH_CHECKPOINTS = PATH_MODELS + "checkpoints/"

MAX_FEATURES = 9358
EMBED_DIM = 300
MAXLEN = 302

#Training
BATCH_SIZE = 8
EPOCHS = 3

"""Splitting the dataset"""

train, test = train_test_split(df, random_state=1, test_size=0.10, shuf-
fle=True)
X_train = np.array(train["text"])
Y_train_f4 = np.array(train["fn_4"]).reshape((-1, 1))
Y_train_f5 = np.array(train["fn_5"]).reshape((-1, 1))
X_test = np.array(test["text"])
Y_test_f4 = np.array(test["fn_4"]).reshape((-1, 1))
Y_test_f5 = np.array(test["fn_5"]).reshape((-1, 1))
print(X_train.shape)
print(X_test.shape)

"""Word Embeddings"""

#OneHotEncoding
Y_train_f4 = to_categorical(Y_train_f4)
Y_test_f4 = to_categorical(Y_test_f4)
Y_train_f5 = to_categorical(Y_train_f5)
Y_test_f5 = to_categorical(Y_test_f5)

#Text to list of indices representing words in dict
tokenizer = Tokenizer(lower=True, split=" ", num_words=MAX_FEATURES)
tokenizer.fit_on_texts(X_train)

X_train_vec = tokenizer.texts_to_sequences(X_train)
X_test_vec = tokenizer.texts_to_sequences(X_test)

MAXLEN = max([len(x) for x in X_train_vec])
print(f"Max vector length: {MAXLEN}")

# pad with zeros for same vector length
X_train_vec = sequence.pad_sequences(X_train_vec, maxlen=MAXLEN, pad-
ding="post")
X_test_vec = sequence.pad_sequences(X_test_vec, maxlen=MAXLEN, pad-
ding="post")

"""FastText"""

#Do onyl once
from gensim.models import KeyedVectors

#Do only once
!wget "https://dl.fbaipublicfiles.com/fasttext/vectors-
crawl/cc.de.300.vec.gz"
!gzip -d cc.de.300.vec.gz
```

```
#Do only once
#Load Fasttext vector embeddings
de_model = KeyedVectors.load_word2vec_format( "cc.de.300.vec")
# use pickle to dump loaded model
pickle.dump(de_model, open("/de_model.pkl", "wb"))
de_model = pickle.load(open("/de_model.pkl", "rb"))

#Loading pickle model
de_model = pickle.load(open("/de_model.pkl", "rb"))

"""Embedding Matrix"""

words_not_found = []
word_index = tokenizer.word_index
nb_words = min(MAX_FEATURES, len(word_index)) +1
# define matrix dimensions
embedding_matrix = np.zeros((nb_words, EMBED_DIM))
for word, i in word_index.items():
    if i >= nb_words:
        continue
    try:
        embedding_vector = de_model.get_vector(word)
    except KeyError:
        embedding_vector = None
    if (embedding_vector is not None) and len(embedding_vector) > 0:
        embedding_matrix[i] = embedding_vector
    else:
        words_not_found.append(word)

"""Model f_4 (emotional empathy)"""

# Define model architecture
from tensorflow.keras.layers import BatchNormalization
model_f4 = Sequential()
model_f4.add(
    Embedding(
        input_dim=nb_words,
        output_dim=EMBED_DIM,
        input_length=MAXLEN,
        weights=[embedding_matrix],
        trainable=True,
    )
)
model_f4.add(LSTM (300,return_sequences=True,dropout=0.80))
model_f4.add(Dense(30,activation='tanh'))
model_f4.add(Flatten())
model_f4.add(Dense(20,activation='relu'))
model_f4.add(Dense(4,activation='softmax'))
model_f4.compile(
    loss="categorical_crossentropy",
    optimizer=tf.keras.optimizers.Adam(),#RMSprop(),
    metrics=["accuracy"],
)
model_f4.summary()

#Training f_5 Model

 %%time
 now = datetime.now().strftime("%Y-%m-%d_%H%M")
 callbacks = [
     EarlyStopping(monitor="val_loss", verbose=1, patience=2),
     ModelCheckpoint(
```

```
        PATH_CHECKPOINTS + now + "_Model_FT-Em-
bed_{epoch:02d}_{val_loss:.4f}.h5",
        monitor="val_loss",
        save_best_only=True,
        verbose=1,
    ),
]

#Fitting the model
steps_per_epoch = int(np.floor((len(X_train_vec) / BATCH_SIZE)))
print(
    f"Model Params.\nbatch_size: {BATCH_SIZE}\nEpochs: {EPOCHS}\n"
    f"Step p. Epoch: {steps_per_epoch}\n"
)

hist = model_f4.fit(
    X_train_vec,
    Y_train_f4,
    batch_size=BATCH_SIZE,
    epochs=EPOCHS,
    steps_per_epoch=steps_per_epoch,
    callbacks=callbacks,
    validation_data=(X_test_vec, Y_test_f4),
)


#Evaluation f_4
pred = model_f4.predict(X_train_vec)
print('Accuracy of f_4 model on Training set')
print(accuracy_score(Y_train_f4.argmax(axis=1), pred.argmax(axis=1)))
print()

# Predict on test data
pred = model_f4.predict(X_test_vec)

# Show prediction metrics
print('Accuracy of f_4 model on Test set')
print(accuracy_score(Y_test_f4.argmax(axis=1), pred.argmax(axis=1)))
print()
print('Confusion Matrix')
print(confusion_matrix(Y_test_f4.argmax(axis=1), pred.argmax(axis=1)))
print()
print('Classification Report')
report = metrics.classification_report(Y_test_f4.argmax(axis=1),
pred.argmax(axis=1))
print(report)

#Saving the model
model_f4.save('/content/drive/My Drive/Data/LSTM/saved models/emotionalem-
pathy')

"""# Model f_5 (cognitive empathy)"""

# Define model architecture

model_f5 = Sequential()
model_f5.add(
    Embedding(
        input_dim=nb_words,
        output_dim=EMBED_DIM,
        input_length=MAXLEN,
        weights=[embedding_matrix],
```

```
        trainable=True,
    )
)

model_f5.add(LSTM (300,return_sequences=True,dropout=0.80))
model_f5.add(Dense(30,activation='tanh'))
model_f5.add(Flatten())
model_f5.add(Dense(20,activation='relu'))
model_f5.add(Dense(4,activation='softmax'))
model_f5.compile(
    loss="categorical_crossentropy",
    optimizer=tf.keras.optimizers.Adam(),#RMSprop(),
    metrics=["accuracy"],
)
model_f5.summary()

#Training f_5 Model

 %%time
 now = datetime.now().strftime("%Y-%m-%d_%H%M")
 callbacks = [
     EarlyStopping(monitor="val_loss", verbose=1, patience=2),
     ModelCheckpoint(
         PATH_CHECKPOINTS + now + "_Model_FT-Em-
bed_{epoch:02d}_{val_loss:.4f}.h5",
         monitor="val_loss",
         save_best_only=True,
         verbose=1,
     ),
 ]

#Fitting the model
 steps_per_epoch = int(np.floor((len(X_train_vec) / BATCH_SIZE)))
 print(
     f"Model Params.\nbatch_size: {BATCH_SIZE}\nEpochs: {EPOCHS}\n"
     f"Step p. Epoch: {steps_per_epoch}\n"
 )

 hist = model_f5.fit(
     X_train_vec,
     Y_train_f5,
     batch_size=BATCH_SIZE,
     epochs=EPOCHS,
     steps_per_epoch=steps_per_epoch,
     callbacks=callbacks,
     validation_data=(X_test_vec, Y_test_f5),
 )


#Evaluation f_5
pred = model_f5.predict(X_train_vec)
print('Accuracy of f_5 model on Training set')
print(accuracy_score(Y_train_f5.argmax(axis=1), pred.argmax(axis=1)))
print()

# Predict on test data
pred = model_f5.predict(X_test_vec)

# Show prediction metrics
print('Accuracy of f_5 model on Test set')
print(accuracy_score(Y_test_f5.argmax(axis=1), pred.argmax(axis=1)))
print()
```

```
print('Confusion Matrix')
print(confusion_matrix(Y_test_f5.argmax(axis=1), pred.argmax(axis=1)))
print()
print('Classification Report')
report = metrics.classification_report(Y_test_f5.argmax(axis=1),
pred.argmax(axis=1))
print(report)

#Saving the model
model_f5.save('/content/drive/My Drive/Data/LSTM/saved models/cognitiveem-
pathy')

"""Loading models"""

#Loading f_4
model_f4=tf.keras.models.load_model('/content/drive/My
Drive/Data/LSTM/saved models/emotionalempathy')
model_f4.summary()

#Predicting
pred = model_f4.predict(X_test_vec)

print('Accuracy of f_4 model on Test set')
print(accuracy_score(Y_test_f4.argmax(axis=1), pred.argmax(axis=1)))

#Loading f_5
model_f5=tf.keras.models.load_model('/content/drive/My
Drive/Data/LSTM/saved models/cognitiveempathy')
model_f5.summary()

#Predicting
pred = model_f5.predict(X_test_vec)

print('Accuracy of f_5 model on Test set')
print(accuracy_score(Y_test_f5.argmax(axis=1), pred.argmax(axis=1)))
```

FARM

```
Set-Up
#installing FARM
!git clone https://github.com/deepset-ai/FARM.git
!pip install -r FARM/requirements.txt
!pip install FARM/

#importing modules
import torch
from farm.modeling.tokenization import Tokenizer
from farm.data_handler.processor import TextClassificationProcessor
from farm.data_handler.data_silo import DataSilo
from farm.modeling.language_model import LanguageModel
from farm.modeling.prediction_head import MultiLabelTextClassificationHead
from farm.modeling.adaptive_model import AdaptiveModel
from farm.modeling.optimization import initialize_optimizer
from farm.infer import Inferencer
from farm.train import Trainer
from farm.utils import MLFlowLogger, initialize_device_settings,
set_all_seeds, MLFlowLogger
import logging
import pandas as pd
from sklearn.model_selection import train_test_split, KFold
import numpy as np

#Tracking the results
#ml_logger = MLFlowLogger(tracking_uri="https://public-mlflow.deepset.ai/")
#ml_logger.init_experiment(experiment_name="Empathy", run_name="final")

#Fetch the right device
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
print("Devices available: {}".format(device))

"""Data Preprocessing"""

#Initializing parameters
set_all_seeds(seed=42)
device, n_gpu = initialize_device_settings(use_cuda=True)
n_epochs = 3
learning_rate = 3e-5
embeds_dropout_prob = 0.1
batch_size = 8
evaluate_every = 100

lang_model = "bert-base-german-cased"
do_lower_case = False

#Loading the dataset
df = pd.read_csv("/Data/dataset_groupedempathylevel.csv")
df.drop(df[df['length']<=3].index, inplace = True)  #droping all rows that
are smaller/equal 3 in length
columns_to_keep = ['text','classID', 'f_4', 'f_5'] #dropping the rest
df = df[columns_to_keep]
df

#Splitting the dataset to train/test
from numpy.random import RandomState
rng = RandomState()
components_train = df.sample(frac=0.8, random_state=42)
components_test = df.loc[~df.index.isin(components_train.index)]
components_train.to_csv('/Data/Farm/train.tsv', sep='\t', index=False,
```

```
header=True)
components_test.to_csv('/Data/Farm/test.tsv', sep='\t', index=False,
header=True)

#Creating a tokenizer (here: BERT tokenizer loaded with german model)
tokenizer = Tokenizer.load(
    pretrained_model_name_or_path=lang_model,
    do_lower_case=do_lower_case)

"""Model Components"""

#Creating a processor to handle conversion from raw text to PyTorch Dataset
label_list = ['strength', "weakness", "suggestions", "None"] #labels in the
data set
metric = "acc_and_f1"  # desired metric for evaluation

processor = TextClassificationProcessor(tokenizer=tokenizer,
                                        max_seq_len=512, # BERT can
only handle sequence lengths of up to 512
                                        data_dir='/Data/Farm',
                                        label_list=label_list,
                                        label_column_name="classID",
                                        metric=metric,
                                        quote_char='"',
                                        multilabel=True,
                                        train_filename="train.tsv",
                                        dev_filename=None,
                                        test_filename="test.tsv",
                                        dev_split=0.2 # this will ex-
tract 20% of the train set to create a dev set
                                       )

#Creating a DataSilo to load various datasets(train/test/dev)
data_silo = DataSilo(
    processor=processor,
    batch_size=batch_size)

#Loading the pretrained BERT german model
language_model = LanguageModel.load(lang_model)

#Define a prediction head that fits for text classification with multiple
labels
prediction_head = MultiLabelTextClassification-
Head(class_weights=data_silo.calculate_class_weights(task_name="text_clas-
sification"),num_labels=len(label_list))

#Create the model
model = AdaptiveModel(
        language_model=language_model,
        prediction_heads=[prediction_head],
        embeds_dropout_prob=embeds_dropout_prob,
        lm_output_types=["per_sequence"],
        device=device)
model.fit_heads_to_lm()

#Creating the optimizer
model, optimizer, lr_schedule = initialize_optimizer(
    model=model,
    device=device,
    learning_rate=learning_rate,
    n_batches=len(data_silo.loaders["train"]),
    n_epochs=n_epochs)
```

```
#Feeding to the Trainer
trainer = Trainer(
        model=model,
        optimizer=optimizer,
        data_silo=data_silo,
        epochs=n_epochs,
        n_gpu=n_gpu,
        lr_schedule=lr_schedule,
        evaluate_every=evaluate_every,
        device=device)
#Training and growing
model = trainer.train()

#Save the model
save_dir_components = "/Farm/saved_models/components_final"
model.save(save_dir_components)
processor.save(save_dir_components)

"""Model f_4 (emotional empathy)"""

#Creating a processor to handle conversion from raw text to PyTorch Dataset
label_list = ['non-empathic', 'empathic', "neutral", "None"] #labels in the
data set
metric = "acc_and_f1"  # desired metric for evaluation

processor = TextClassificationProcessor(tokenizer=tokenizer,
                                        max_seq_len=512, # BERT can
only handle sequence lengths of up to 512
                                        data_dir='/My Drive/Data/Farm',
                                        label_list=label_list,
                                        label_column_name="f_4",
                                        metric=metric,
                                        quote_char='"',
                                        multilabel=True,
                                        train_filename="train.tsv",
                                        dev_filename=None,
                                        test_filename="test.tsv",
                                        dev_split=0.2 # this will ex-
tract 20% of the train set to create a dev set
                                       )

#Creating a DataSilo to load various datasets(train/test/dev)
data_silo = DataSilo(
    processor=processor,
    batch_size=batch_size)

#Loading the pretrained BERT german model
language_model = LanguageModel.load(lang_model)

#Define a prediction head that fits for text classification with multiple
labels
prediction_head = MultiLabelTextClassification-
Head(class_weights=data_silo.calculate_class_weights(task_name="text_clas-
sification"),num_labels=len(label_list))

#Create the model
model = AdaptiveModel(
        language_model=language_model,
        prediction_heads=[prediction_head],
        embeds_dropout_prob=embeds_dropout_prob,
        lm_output_types=["per_sequence"],
```

```
        device=device)
model.fit_heads_to_lm()

#Creating the optimizer
model, optimizer, lr_schedule = initialize_optimizer(
    model=model,
    device=device,
    learning_rate=learning_rate,
    n_batches=len(data_silo.loaders["train"]),
    n_epochs=n_epochs)

#Feeding to the Trainer
trainer = Trainer(
        model=model,
        optimizer=optimizer,
        data_silo=data_silo,
        epochs=n_epochs,
        n_gpu=n_gpu,
        lr_schedule=lr_schedule,
        evaluate_every=evaluate_every,
        device=device)
#Training and growing
model = trainer.train()

#Save the model
save_dir_f4 = "/Farm/saved_models/emotionalempathy_final"
model.save(save_dir_f4)
processor.save(save_dir_f4)

"""Model f_5 (cognitive empathy)"""

#Creating a processor to handle conversion from raw text to PyTorch Dataset
label_list = ['non-empathic', 'empathic', "neutral", "None"] #labels in the
data set
metric = "acc_and_f1"  # desired metric for evaluation

processor = TextClassificationProcessor(tokenizer=tokenizer,
                                        max_seq_len=512, # BERT can
only handle sequence lengths of up to 512
                                        data_dir='/My Drive/Data/Farm',
                                        label_list=label_list,
                                        label_column_name="f_5",
                                        metric=metric,
                                        quote_char='"',
                                        multilabel=True,
                                        train_filename="train.tsv",
                                        dev_filename=None,
                                        test_filename="test.tsv",
                                        dev_split=0.2 # this will ex-
tract 20% of the train set to create a dev set
                                       )

#Creating a DataSilo to load various datasets(train/test/dev)
data_silo = DataSilo(
    processor=processor,
    batch_size=batch_size)

#Loading the pretrained BERT german model
language_model = LanguageModel.load(lang_model)

#Define a prediction head that fits for text classification with multiple
labels
```

```
prediction_head = MultiLabelTextClassification-
Head(class_weights=data_silo.calculate_class_weights(task_name="text_clas-
sification"), num_labels=len(label_list))

#Create the model
model = AdaptiveModel(
        language_model=language_model,
        prediction_heads=[prediction_head],
        embeds_dropout_prob=embeds_dropout_prob,
        lm_output_types=["per_sequence"],
        device=device)
model.fit_heads_to_lm()

#Creating the optimizer
model, optimizer, lr_schedule = initialize_optimizer(
    model=model,
    device=device,
    learning_rate=learning_rate,
    n_batches=len(data_silo.loaders["train"]),
    n_epochs=n_epochs)

#Feeding to the Trainer
trainer = Trainer(
        model=model,
        optimizer=optimizer,
        data_silo=data_silo,
        epochs=n_epochs,
        n_gpu=n_gpu,
        lr_schedule=lr_schedule,
        evaluate_every=evaluate_every,
        device=device)

#Training and growing
model = trainer.train()

#Save the model
save_dir_f5 = "/content/drive/My Drive/Data/Farm/saved_models/cognitiveem-
pathy_final"
model.save(save_dir_f5)
processor.save(save_dir_f5)


"""Test on Sample"""

#Test the model on a sample
from farm.infer import Inferencer
from pprint import PrettyPrinter

basic_texts = [{"text": "Das Template wurde gut umgesetzt. Die Darstellung
ist schlüssig, Persona und User Cycle passen zusammen."
}]
#inferenced_model = Inferencer.load(/saved_models/components_final")
inferenced_model= Inferencer.load("/saved_models/cognitiveempathy_final")
#inferenced_model= Inferencer.load("/saved_models/emotionalempathy_final")
result = inferenced_model.inference_from_dicts(dicts=basic_texts)
PrettyPrinter().pprint(result)
```

ELEA – Frontend

```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no">

    <!-- Bootstrap CSS -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/boot-
strap/4.0.0/css/bootstrap.min.css" integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" cros-
sorigin="anonymous">
    <link rel="stylesheet" href="{{ url_for('static', file-
name='css/style.css') }}">
    <link rel="stylesheet" type="text/css" href="{{ url_for('static', file-
name='css/progress.css') }}">
    <title>ELEA</title>
  </head>
  <body>
    <div id="overlay" style="display:none;">
      <div class="spinner"></div>
      <br/>
      <p class="text" style="color: cadetblue;font-weight: bold;">Please be
patient, ELEA is analyzing your text.</p>
    </div>
    <br>
    <div class="container">
        <nav class="navbar navbar-expand navbar-dark sticky-top"
style="background-color: #FFFFFF;">
          <a class="navbar-brand">
            <img src="{{ url_for('static', filename='images/ELEA.png') }}"
alt="Logo" style="height:50px;">
          </a>
          <ul class="navbar-nav mr-auto"></ul>
          <ul class="navbar-nav">
            <li class="nav-item">


              <img src="{{ url_for('static', filename='images/logo-
im.jpg') }}" alt="Logo" style="height:50px;">
            </a>
          </li>
        </ul>
      </nav>
      <div class="jumbotron jumbotron-fluid">
          <div class="container">
            <h5 class="text-center">Please read the following business
idea carefully:</h5>
            <p class="lead text-justify">
                SecondLife ist eine neuartige Schweizer Plattform für
die umweltbewusste, nachhaltig-denkende Person. Die Plattform ermöglicht
Menschen, sich in einer Zeit von Überkonsum und Luxus zurechtzufinden und
gleichzeitig ihren Beitrag an die Umwelt zu leisten, ohne auf Luxus ver-
zichten zu müssen. Jede Person kann sich kostenlos auf der Plattform re-
gistrieren und ein persönliches Konto erstellen. Damit erhält man Zugang zu
einem auserwählten Marktplatz von Designer- und Marken-Fashionartikel mit
speziellem Fokus auf nachhaltiger Produktion. SecondLife wählt ihre Anbie-
ter mit Sorgfalt aus und stellt sicher, dass nur fair produzierte Modearti-
```

```
kel auf die Plattform gelangen. Registrierte Benutzer haben nun die Mög-
lichkeit, bestimmte Kleider und Modeartikel auf dem Marktplatz zu erwerben
oder zu mieten. Da die Kleider auf dem Marktplatz bereits auf Events,
Fashionshows oder im Laden getragen wurden und somit nicht «neu-ab-Stange»
sind, ist der Preis reduziert. SecondLife garantiert mindestens 20% Preis-
nachlass auf den üblichen Verkaufspreis. Der definitive Preis wird vom An-
bieter bestimmt. Nutzer zahlen direkt über die Plattform an den Anbieter,
wobei ein kleiner Betrag für die Weiterentwicklung von SecondLife behalten
wird. Wird ein Kleidungsstück gemietet, kann es kostenlos zum Anbieter zu-
rückgeschickt werden. Kleider, welche mindestens bereits 10x vermietet wur-
den, müssen vom Anbieter zukünftig mit einem Preisnachlass von 50% angebo-
ten werden. SecondLife garantiert einen sicheren und anonymisierten Umgang
mit jeglichen Daten der Anbieter und Nachfrager.
              </p>
            </div>
        </div>
        <hr><h5 class="text-center">Please write your peer review.</h5>

        <form class="text-center" action="{{ url_for('index') }}"
method="POST">
   <p </p>

            <div class="form-group">
              <label for="strengths">Enter the strengths about the peer's
business model</label>
              <textarea class="form-control" name='strengths'
id="strengths" cols="30" rows="5" required>{{re-
quest.form['strengths']}}</textarea>
            </div>
            <br>
            <div class="form-group">
              <label for="weaknesses">Enter the weaknesses about the peer's
business model</label>
              <textarea class="form-control" name='weaknesses' id="weak-
nesses" cols="30" rows="5" required>{{request.form['weaknesses']}}</tex-
tarea>
            </div>
            <br>
            <div class="form-group">
              <label for="improvements">Enter suggestions for improvements
about the peer's business model</label>
              <textarea class="form-control" name='suggestions' id="sugges-
tions" cols="30" rows="5" required>{{request.form['suggestions']}}</tex-
tarea>
            </div>
            <button type="submit" class="btn btn-success btn-lg center"
id="analyze" style="background-color: #00802F;"  onclick="on()">Ana-
lyze</button>
            <br><br><br>
        </form>

        <hr>
        <div class="row mb-5">
          <div class="col-xl-12 rounded border">
            {% if emo_labels and cog_labels %}
              <div id="feedback">
                <div class="container-fluid text-center mt-3">
                  <h1>Your empathy learning dashboard</h1><br/>
                  <p class="text"<^>If you want to know more about how ELEA
works, click <a href="#" onclick="window.open('{{ url_for('popup') }}',
'ELEA', 'width=500,height=500');" >here</a>.</p>
```

```
                </div>

                <!-- DASHBORAD STATS -->

                <div class="row text-center" style="height: auto">
                  <div class="col-md-6 text-left ml-5">
                    <h4>Detailed feedback on your review</h4>
                    <br>
                      <h5>Strengths</h5>
                      {% if 'None' not in emo_labels['strength'] %}
                      <p class="text"><i>Emotional Empathy:</i> Your input
text was <span style="color: orange;">{{ emo_la-
bels['strength'].split("'")[1] }}.</span></p>
                      <p class="text text-justify">{{ emo_feed-
back['strength'][emo_labels['strength'].split("'")[1]] }}</p>
                      {% else %}
                        No label was predicted. Please re-enter your feed-
back.<br>
                      {% endif %}
                      {% if 'None' not in cog_labels['strength'] %}
                      <p class="text"><i>Cognitive Empathy:</i> Your input
text was <span style="color: orange;">{{ cog_la-
bels['strength'].split("'")[1] }}.</span></p>
                      <p class="text text-justify">{{ cog_feed-
back['strength'][cog_labels['strength'].split("'")[1]] }}</p>
                      {% else %}
                        No label was predicted. Please re-enter your feed-
back.<br>
                      {% endif %}
                      <h5>Weaknesses</h5>
                      {% if 'None' not in emo_labels['weakness'] %}
                      <p class="text"><i>Emotional Empathy:</i> Your input
text was  <span style="color: orange;">{{ emo_labels['weak-
ness'].split("'")[1] }}.</span></p>
                      <p class="text text-justify">{{ emo_feedback['weak-
ness'][emo_labels['weakness'].split("'")[1]] }}</p>
                      {% else %}
                        No label was predicted. Please re-enter your feed-
back.<br>
                      {% endif %}
                      {% if 'None' not in cog_labels['weakness'] %}
                      <p class="text"><i>Cognitive Empathy:</i> Your input
text was  <span style="color: orange;">{{ cog_labels['weak-
ness'].split("'")[1] }}.</span></p>
                      <p class="text text-justify">{{ cog_feedback['weak-
ness'][cog_labels['weakness'].split("'")[1]] }}</p>
                      {% else %}
                        No label was predicted. Please re-enter your feed-
back.<br>
                      {% endif %}
                      <h5>Suggestions for Improvements</h5>
                      {% if 'None' not in emo_labels['suggestion'] %}
                      <p class="text"><i>Emotional Empathy:</i> Your input
text was  <span style="color: orange;">{{ emo_labels['sugges-
tion'].split("'")[1] }}.</span></p>
                      <p class="text text-justify">{{ emo_feedback['sugges-
tion'][emo_labels['suggestion'].split("'")[1]] }}</p>
                      {% else %}
                        No label was predicted. Please re-enter your feed-
back.<br>
                      {% endif %}
                      {% if 'None' not in cog_labels['suggestion'] %}
```

```
                        <p class="text"><i>Cognitive Empathy:</i> Your input
text was  <span style="color: orange;">{{ cog_labels['sugges-
tion'].split("'")[1] }}.</span></p>
                        <p class="text text-justify">{{ cog_feedback['sugges-
tion'][cog_labels['suggestion'].split("'")[1]] }}</p>
                        {% else %}
                          No label was predicted. Please re-enter your feed-
back.<br>
                        {% endif %}


                    </div>
                    <div class="col-md-5">
                      <h4>General Overview</h4>
                      <br>
                        <div class="progress mx-auto" data-value='{{
perc_score }}'>
                          <span class="progress-left">
                              <span class="progress-bar border-suc-
cess"></span>
                          </span>
                          <span class="progress-right">
                              <span class="progress-bar border-suc-
cess"></span>
                          </span>
                          <div class="progress-value w-100 h-100 rounded-cir-
cle d-flex align-items-center justify-content-center">
                            <div class="h2 font-weight-bold">{{ perc_score
}}</div><sup class="small">%</sup>
                          </div>
                        </div>
                        <br><br>
                        <p class="text">Empathy Score: {{ perc_score }}%</p>
                        <p class="text" style="color:orange; font-weight:
600;">
                          {{ feedback }}
                        </p>
                      </div>
                    </div>
                    {% endif %}

                </div>

            </div>
        </div>


    </div>
    <!-- Optional JavaScript -->
    <!-- jQuery first, then Popper.js, then Bootstrap JS -->
    <script src="https://ajax.goog-
leapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
    <script type="text/javascript">
      $(function() {

        $(".progress").each(function() {

          var value = $(this).attr('data-value');
          var left = $(this).find('.progress-left .progress-bar');
          var right = $(this).find('.progress-right .progress-bar');

          if (value > 0) {
```

```
            if (value <= 50) {
              right.css('transform', 'rotate(' + percentageToDegrees(value)
+ 'deg)')
            } else {
              right.css('transform', 'rotate(180deg)')
              left.css('transform', 'rotate(' + percentageToDegrees(value -
50) + 'deg)')
            }
          }

        })

        function percentageToDegrees(percentage) {
          return percentage / 100 * 360
        }

        });

        document.getElementById('feedback').scrollIntoView();
        event.preventDefault();
    </script>

    <script>
      // function on() {
      //   $.LoadingOverlay("show");
      // }
      $(document).ready(function() {
              $('#analyze').click(function(){
                if ($('#strengths').val() != '' && $('#weaknesses').val()
!= '' && $('#suggestions').val() != ''){
                  $('#overlay').fadeIn().delay(100000).fadeOut();
              }
            });


        });
    </script>
    <script>
      if ( window.history.replaceState ) {
          window.history.replaceState( null, null, window.location.href );
      }
    </script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/pop-
per.js/1.12.9/umd/popper.min.js" integrity="sha384-Ap-
Nbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q" cros-
sorigin="anonymous"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/boot-
strap.min.js" integrity="sha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl" cros-
sorigin="anonymous"></script>

  </body>
</html>
```

ELEA-Backend

```
from flask import Flask, redirect, render_template, request, session,
url_for
import torch
torch.multiprocessing.freeze_support()
from farm.infer import Inferencer
import gc

model_dir_cog = "cognitiveempathy"
model_dir_emo = "emotionalempathy"

app = Flask(__name__)

def inference_cognitive(inferencer, basic_texts):
    result = inferencer.inference_from_dicts(dicts=basic_texts)
    label = result[0]['predictions'][0]['label']
    return label.split(',')[0]

def inference_emotional(inferencer, basic_texts):
    result = inferencer.inference_from_dicts(dicts=basic_texts)
    label = result[0]['predictions'][0]['label']
    return label.split(',')[0]

def calculate_empathy_score(emo_labels, cog_labels):
    score = 0
    for label in emo_labels.values():
        if 'empathic' in label and 'non' not in label:
            score += 3
        elif 'non-empathic' in label:
            score += 1
        elif 'neutral' in label:
            score += 2
        else:
            score += 0

    for label in cog_labels.values():
        if 'empathic' in label and 'non' not in label:
            score += 3
        elif 'non-empathic' in label:
            score += 1
        elif 'neutral' in label:
            score += 2
        else:
            score +=0

    # print(score)
    percentage_score = (score / 18) * 100
    return int(percentage_score)

def get_feedback(score):
    feedback = ''
    if score > 0 and score <= 20:
        feedback = '''Very weak: Your feedback still lacks a lot of empa-
thy. Try to include more emotional and cognitive aspects in your review of
the peer's business idea.'''
    elif score > 20 and score <= 40:
        feedback = '''Weak: Your feedback is still missing a lot of em-
pathic aspects. Try to add more emotional feelings and step further into
your peer's perspective.'''
    elif score > 40 and score <= 60:
        feedback = '''Neutral. Your feedback is written very objectively.
```

```
Try to include more of your personal thoughts and add further explanations,
elaborations and personal feelings to your review.'''
    elif score > 60 and score <= 80:
        feedback = '''Good. Your feedback shows a good level of empathy.
You managed to include personal feelings and step into the peer's perspec-
tive to elaborate fully on the business idea. Try to add a few further
elaborations and personal feelings to your review.'''
    elif score > 80 and score <= 100:
        feedback = "Very well. Your feedback is very empathic!"

    return feedback

def get_cognitive_feedback():
    cognitive_feedback_texts = {
        'strength':{
            'empathic': "Well done, you managed to step outside your own
perspective and think from the peer’s perspective. Moreover, you review in-
cludes details, explanations, questions, or direct addressing of the author
to better understand and elaborate on the peer’s perspective.",
            'non-empathic':"Your feedback is very short and does not in-
clude the peer’s perspective. Try to step into his shoes and add examples,
explanations or further elaborations to your feedback. ",
            'neutral':"Try to add more contextual rather than formal as-
pects of the peer’s business idea and add further explanations and elabora-
tions on your thoughts by thinking from the peer’s perspective. "
        },
        'weakness':{
            'empathic': "Well done, you managed tostep outside your own
perspective and think from the peer’s perspective. Moreover, you review in-
cludes details, explanations, questions, or direct addressing of the author
to better understand and elaborate on the peer’s perspective.",
            'non-empathic':"Your feedback is very short and does not in-
clude the peer’s perspective. Try to step into his shoes and add examples,
explanations or further elaborations to your feedback. Explain, why the
mentioned weakness is important to consider.",
            'neutral':"Try to add more contextual rather than formal as-
pects of the peer’s business idea and add further explanations and elabora-
tions on your thoughts by thinking from the peer’s perspective. Explain,
why the mentioned weakness is important to consider. "
        },
        'suggestion'{
            'empathic': "Well done, you managed to step outside your own
perspective and thinks from the peer’s perspective. Moreover, you review
includes details, explanations, questions, or direct addressing of the au-
thor to better understand and elaborate on the peer’s perspective.",
            'non-empathic':"Your feedback is very short and does not in-
clude the peer’s perspective. Try to step into his shoes and add examples,
explanations or further elaborations to your feedback. Be very specific on
your instructions for improvements and explain them in detail.",
            'neutral':"Try to add more contextual rather than formal as-
pects of the peer’s business idea and add further explanations and elabora-
tions on your thoughts by thinking from the peer’s perspective. Be very
specific on your instructions for improvements and explain them in detail.
"}}
    return cognitive_feedback_texts

def get_emotional_feedback():
    emotional_feedback_texts = {
        'strength': {
            'empathic': "Well done, you managed to show your own personal
feelings and emotions towards the peer’s business idea",
            'non-empathic':"Try to respond in an emotional manner to your
```

```
peer's business idea by including your personal feelings and emotions.
Share your excitement for the strengths you have detected",
            'neutral':"You already used emotions in your feedback text, but
you are still missing to state your own personal feelings by using personal
pronouns when describing the strengths. "
        },
        'weakness':{
            'empathic': "Well done, you managed to show your own personal
feelings and emotions towards the peer's business idea.",
            'non-empathic':"Try to respond in an emotional manner to your
peer's business idea by including your personal feelings and emotions.
Share your personal concerns or doubts for the weaknesses you have de-
tected. ",
            'neutral':"You already used emotions in your feedback text, but
you are still missing to state your own personal feelings by using personal
pronouns when describing the weakness. "
        },
        'suggestion':{
            'empathic': "Well done, you managed to show your own personal
feelings and emotions towards the peer's business idea. ",
            'non-empathic':"Try to respond in an emotional manner to your
peer's business idea by including your personal feelings and emotions. ",
            'neutral':"You already used emotions in your feedback text, but
you are still missing to state your own personal feelings by using personal
pronouns when describing your suggestions for improvement. "}}

    return emotional_feedback_texts



@app.route('/', methods=['GET', 'POST'])
def index():
    inp1 = {}
    inp2 = {}
    inp3 = {}

    if request.method == 'POST':
        cog_labels = {}
        emo_labels = {}

        cognitive_inferencer= Inferencer.load(model_dir_cog)
        emotional_inferencer= Inferencer.load(model_dir_emo)

        inp1['text'] = request.form['strengths']
        inp2['text'] = request.form['weaknesses']
        inp3['text'] = request.form['suggestions']

        cog_labels['strength'] = inference_cognitive(cognitive_inferencer,
[inp1])
        emo_labels['strength'] = inference_emotional(emotional_inferencer,
[inp1])

        cog_labels['weakness'] = inference_cognitive(cognitive_inferencer,
[inp2])
        emo_labels['weakness'] = inference_emotional(emotional_inferencer,
[inp2])

        cog_labels['suggestion'] = inference_cognitive(cognitive_infer-
encer, [inp3])
        emo_labels['suggestion'] = inference_emotional(emotional_infer-
encer, [inp3])
```

```
        percentage_score = calculate_empathy_score(emo_labels, cog_labels)

        feedback = get_feedback(percentage_score)

        del cognitive_inferencer
        del emotional_inferencer

        return render_template('index.html',
                               perc_score=percentage_score,
                               feedback=feedback,
                               emo_labels=emo_labels,
                               cog_labels=cog_labels,
                               emo_feedback=get_emotional_feedback(),
                               cog_feedback=get_cognitive_feedback(),)
    return render_template('index.html', onclick = "popup.html")

@app.route('/popup')
def popup():
    return render_template('popup.html')


if __name__ == '__main__':
    app.secret_key = 'you shall not guess'
    #app.run(threaded=True)
    app.run(host='0.0.0.0', port=5000, threaded=True)
```

Control Group – Dictionary-based approach

Keywords were selected based on a study with the human annotators. Each annotator was asked to select a list of words or phrases from the peer reviews he/she annotated, that represented high emotional or cognitive empathy.

```
from flask import Flask, redirect, render_template, request, session,
url_for
import gc

app = Flask(__name__)

keywords = ['grundsätzlich', "allgemein", "beispielsweise", "zum beispiel",
"wünschen", "mein", "meine", "meiner", "mir", "dir", "ich", "du", "mein-
ung", "interessant", "wichtig", "kritisch", "persönlich", "sehr",
"äusserst", "gut", "super", "toll", "top", "stark", "spannend",
"aussergewöhnlich", "herausragend", "gelungen", "positiv", "negativ",
"schlecht", "kritisch", "unsicher"]

def dictionary_score(text_strength, text_weakness, text_suggestion):
    wordcount = {'strength': 0, 'weakness': 0, 'suggestion': 0}

    words_strength = text_strength.strip().lower().split(" ")
    words_weakness = text_weakness.strip().lower().split(" ")
    words_suggestion = text_suggestion.strip().lower().split(" ")

    for word in words_strength:
        if word in keywords:
            wordcount['strength'] += 1
    for word in words_weakness:
        if word in keywords:
            wordcount['weakness'] += 1
    for word in words_suggestion:
```

```
        if word in keywords:
            wordcount['suggestion'] += 1
    print
    return wordcount

def total_score(wordcount):
    total = 0
    for score in wordcount.values():
        total  += int(score)
    return total

def get_feedback(total_score):
    feedback = ''
    if total_score > 0 and total_score  <= 10:
        feedback = '''Try to use more keywords such as "in my opinion,", "I
feel", "I think", "Your idea is great", etc. to make your text more em-
pathic.'''
    elif total_score > 10:
        feedback = '''Try to use more keywords such as "in my opinion,", "I
feel", "I think", "Your idea is great", etc. to make your text more em-
pathic.'''

    return feedback


@app.route('/', methods=['GET', 'POST'])
def index():
    show_feedback = False
    if request.method == 'POST':

        inp1 = request.form['strengths']
        inp2 = request.form['weaknesses']
        inp3 = request.form['suggestions']


        wordcount = dictionary_score(inp1, inp2, inp3)

        totalscore = total_score(wordcount)

        feedback = get_feedback(totalscore)

        show_feedback = True



        # return redirect(url_for('index', perc_score=percentage_score,
feedback=feedback, emo_labels=emo_labels, cog_labels=cog_labels))
        return render_template('index.html',
                               wordcount = wordcount,
                               total_score = totalscore,
                               feedback = feedback,
                               show_feedback = show_feedback
                               )
    return render_template('index.html')

@app.route('/popup')
def popup():
    return render_template('popup.html')

if __name__ == '__main__':
    app.secret_key = 'you shall not guess'
```

```
app.run(threaded=True)
#app.run(host='0.0.0.0', port=5000, threaded=True)
```

C: Data Analysis

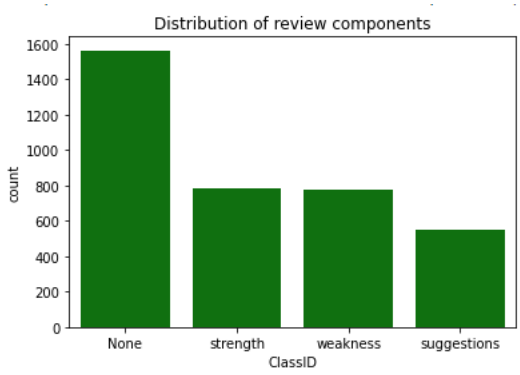


Figure 20: Distribution of review components (own illustration)

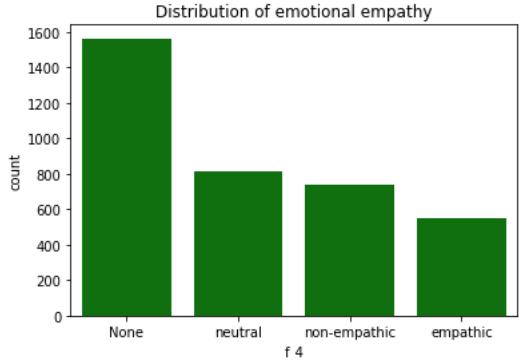


Figure 21: Distribution of emotional empathy level (own illustration)

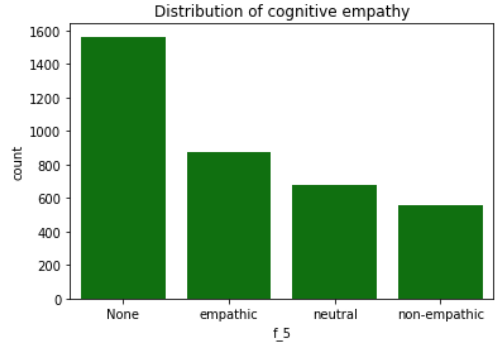


Figure 22: Distribution of cognitive empathy level (own illustration)