UNIVERSITY OF POTSDAM

MASTER'S THESIS

SoPa++: Leveraging explainability from hybridized RNN, CNN and weighted finite-state neural architectures

Author:
Atreya SHANKAR

1st Supervisor:
Dr. Sharid LOÁICIGA
University of Potsdam

2nd Supervisor:
Mathias MÜLLER
University of Zurich

A thesis submitted in fulfillment of the requirements for the degree of Cognitive Systems: Language, Learning, and Reasoning (M.Sc.)

in the

Foundations of Computational Linguistics Research Group
Department of Linguistics

March 15, 2021

Contents

Chapter 1

Introduction

1.1 Motivation

With the recent trend of increasingly large deep learning models achieving State-Of-The-Art (SOTA) performance on a myriad of Machine Learning (ML) tasks (Figure 1), several studies argue for focused research into Explainable Artificial Intelligence (XAI) to address emerging concerns such as security risks and inductive biases associated with black-box models (Doran, Schulz, and Besold, 2017; Townsend, Chaton, and Monteiro, 2019; Danilevsky et al., 2020; Arrieta et al., 2020). Of these studies, Arrieta et al. (2020, Section 2.2, Page 4) provide the following novel definition of XAI based on an extensive literature review of recent XAI research:

> *"Given an audience, an **explainable** Artificial Intelligence is one that produces details or reasons to make its functioning clear or easy to understand."*

In addition, Arrieta et al. (2020) explore and classify a variety of machine-learning models into transparent and black-box categories depending on their degrees of transparency. Furthermore, they explore taxonomies of post-hoc explainability methods aimed at effectively explaining black-box models. Of high relevance to this study are the local explanations, feature relevance and explanations by simplification post-hoc explainability techniques.

Through a survey of recent literature on explanations by simplification applied in the Natural Language Processing (NLP) field, we came across several prominent studies employing techniques to simplify black-box neural networks into constituent Finite-State Automata (FSAs) and/or Weighted Finite-State Automata (WFSAs) (Schwartz, Thomson, and Smith, 2018; Peng et al., 2018; Suresh et al., 2019; Wang and Niepert, 2019; Jiang et al., 2020).

In this thesis, we build upon the work of Schwartz, Thomson, and Smith (2018) by further developing their **So**ft **Pa**tterns (SoPa) model; which represents a hybridized RNN, CNN and Weighted Finite-State Automaton neural network architecture. We modify the SoPa model by changing key aspects of its architecture which ultimately allows us to conduct effective explanations by simplification; which was not possible with the previous SoPa architecture. We abbreviate this modified model as **SoPa++**, which signifies an improvement or major modification to the SoPa model. Finally, we evaluate both the performance and explainability of the SoPa++ model on the Facebook Multilingual Task Oriented Dialog data set (FMTOD; Schuster et al. 2019); focusing on the English-language intent classification task.

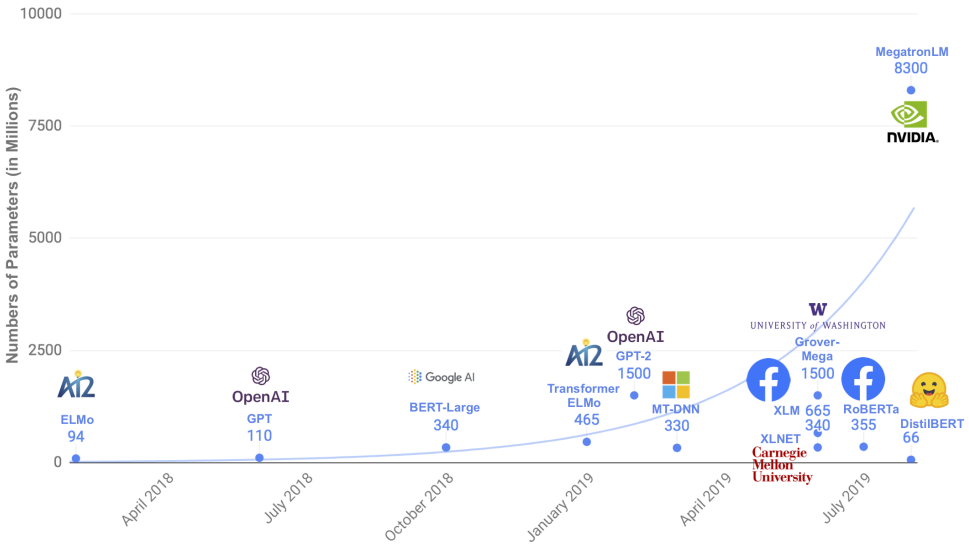

FIGURE 1: Parameter counts of recently released pre-trained language models which showed competitive or SOTA performance when fine-tuned over a range of NLP tasks; figure taken from Sanh et al. (2019)

1.2 Research questions

With the aforementioned modifications to the SoPa architecture and the introduction of the SoPa++ architecture, we aim to answer the following three research questions:

1. To what extent does SoPa++ contribute to competitive performance[1] on the FMTOD data set?
2. To what extent does SoPa++ contribute to effective explanations by simplification on the FMTOD data set?
3. What interesting and relevant explanations can SoPa++ provide on the FMTOD data set?

1.3 Thesis structure

With the aforementioned research questions, we summarize the structure and contents of this thesis.

Chapter 1: Introduce this thesis, its contents and our research questions.

Chapter 2: Describe the background concepts utilized in this thesis.

Chapter 3: Describe the methodologies pursued in this thesis.

Chapter 4: Describe the results obtained from our methodologies.

Chapter 5: Discuss the implications of the aforementioned results.

Chapter 6: Conclude this thesis by answering the research questions.

Chapter 7: Document future work to expand on our research questions.

[1] We define competitive performance as the scenario where a mean performance metric on a certain data set falls within the range obtained from other recent studies on the same data set

Chapter 2

Background concepts

2.1 Explainable artificial intelligence

In this section, we lay out background concepts for Explainable Artificial Intelligence (XAI) which have been largely adopted from Arrieta et al. (2020). The study is particularly helpful for us since it summarizes the findings of approximately 400 XAI contributions and presents these findings in the form of well-defined concepts and taxonomies. In addition, the study discusses future directions of XAI research. We start off by providing definitions from the study, along with accompanying remarks.

2.1.1 Transparency

Definition 1 (Transparency; Arrieta et al. 2020)**.** A model is considered to be transparent if it is understandable on its own without any external techniques to assist its understandability. Since a model can provide different degrees of understandability, transparent models are split into three possibly mutually-inclusive categories; specifically simulatable models, decomposable models and algorithmically transparent models.

Remark 1.1. *Simulatability* refers to the ability of a model's inner-mechanisms being simulated strictly by a human. Therefore, model simplicity plays a major role in this class.

Remark 1.2. *Decomposability* refers to the ability to clearly understand the individual parts of a model; such as its inputs, parameters and basic computational mechanisms.

Remark 1.3. *Algorithmic transparency* refers to the ability of a human to understand the algorithmic processes used in the model to produce any given output from any given input.

Remark 1.4. A model is considered transparent if it falls into one or more of the aforementioned transparency categories. If a model cannot satisfy any of the requirements of being transparent, then it is classified as a *black-box* model.

Remark 1.5. As recommended by Arrieta et al. (2020, Section 2.1, Page 3), we use the terms transparency and interpretability to refer to the same feature. Therefore, we use these terms equivalently and interchangeably.

Examples of well-known transparent Machine Learning (ML) models are linear/logistic regressors, decision trees and rules-based learners. Similarly, common examples of black-box ML models are tree ensembles and deep neural networks. Arrieta et al. (2020) provide extensive justifications using the aforementioned three criteria in conducting model classifications into the transparent and black-box categories. We would direct the reader to their study for a full analysis and justification of these classifications.

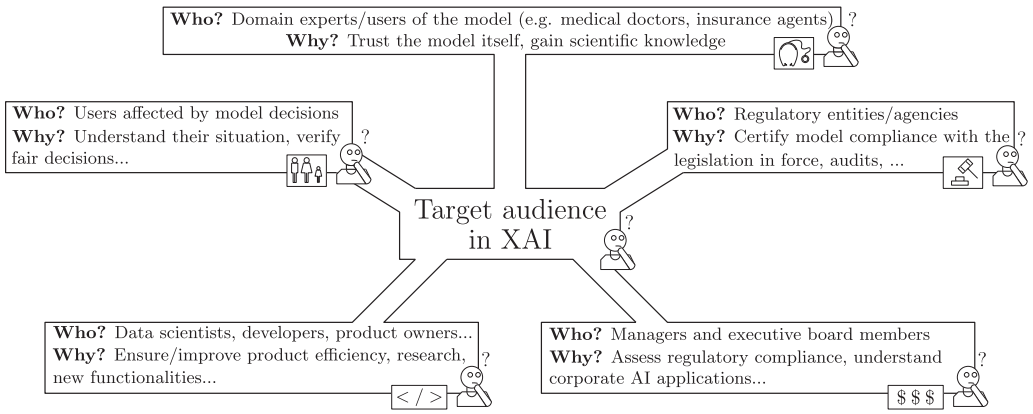

FIGURE 2: Examples of various target audiences in XAI; figure taken from Arrieta et al. (2020)

2.1.2 Explainability and XAI

Definition 2 (Explainability; Arrieta et al. 2020)**.** Explainability refers to the interface between humans and a model, such that this interface is both an accurate proxy of the model and comprehensible to humans.

Definition 3 (Explainable Artificial Intelligence; Arrieta et al. 2020)**.** An explainable artificial intelligence is one that produces details to make its functioning understandable to a given target audience.

Arrieta et al. (2020) observe that black-box ML models are increasingly being employed to make important predictions in critical contexts, citing high-risk areas such as precision medicine and autonomous vehicles. Of particular relevance to the field of Natural Language Processing (NLP), the study notes a myriad of issues related to inductive biases within training data sets and the ethical issues involved in using black-box models trained on such data sets. As a result, they describe an increased demand for transparency in black-box ML models from the various stakeholders in Artificial Intelligence (AI). In addition, Arrieta et al. (2020) emphasize the presence of a target audience for XAI; implying that different XAI techniques should be employed for different target audiences. In their study, they provide examples of target audiences such as domain experts, end-users and managers (Figure 2).

2.1.3 Terminology clarification

Based on an extensive literature review, Arrieta et al. (2020) observe that many studies tend to misuse the terms interpretability and explainability by using them interchangeably. To address this, they provide clear conceptual differences between the terms. For example, Arrieta et al. (2020, Section 2.1, Page 3) state that *"interpretability refers to a passive characteristic of a model referring to the level at which a given model makes sense for a human observer."* In contrast, Arrieta et al. (2020, Section 2.1, Page 3) state that *"explainability can be viewed as an active characteristic of a model, denoting any action or procedure taken by a model with the intent of clarifying or detailing its internal functions."*

In summary, we gather that interpretability (or transparency as per Remark 1.5) refers to an inherent or passive feature of a model. On the other hand, explainability refers to an active characteristic undertaken by the model and its developers to explain the model's inner mechanisms. In addition, explainability entails the presence

of a target audience; which may not necessarily be the case for interpretability or transparency.

2.1.4 Explainability techniques

Based on the aforementioned classification of ML models into transparent and black-box models, Arrieta et al. (2020) expound on explainability techniques for each of these model types. Due to their transparent nature, the study states that transparent ML models are usually explainable in themselves to most target audiences and therefore usually do not require any external technique to extract explanations. The study does however highlight some target audiences, such as non-expert users, who may require external explainability techniques such as model output visualizations in order to explain the inner workings of transparent ML models.

For the case of black-box models, Arrieta et al. (2020) argue that separate or external techniques must be utilized in order to reasonably explain these models. Such explainability techniques are referred to in the study as post-hoc explainability techniques; which is derived from the idea that explanations for such models are usually extracted post-modeling. Notable examples of post-hoc explainability techniques include local explanations, feature relevance and explanations by simplification.

Definition 4 (Local explanations; Arrieta et al. 2020)**.** Local explanations operate by segmenting a model's solution space into subspaces and provide explanations for the less complex model subspaces.

Remark 4.1. Local explanations are commonly used in XAI research and function by using differentiating properties on model solution space subsets.

Remark 4.2. Two well-known examples of local explainability techniques are Local Interpretable Model-Agnostic Explanations (LIME; Ribeiro, Singh, and Guestrin 2016) and G-REX (Konig, Johansson, and Niklasson, 2008).

Definition 5 (Feature relevance; Arrieta et al. 2020)**.** Feature relevance explanation methods operate by computing an importance score for the model's input variables over the model's output variables. These scores typically quantify how sensitive the model's output is to perturbations in the model's inputs.

Remark 5.1. Feature relevance methods can be considered as indirect methods of explaining a model.

Remark 5.2. Well-known feature relevance explainability techniques include the Shapley Additive Explanations (SHAP; Lundberg and Lee 2017) and the occlusion sensitivity method (Zeiler and Fergus, 2014).

Definition 6 (Explanations by simplification; Arrieta et al. 2020)**.** Explanations by simplification refer to techniques where a simplified proxy model is built to approximate and explain a more complex model. The simplified proxy model usually has to fulfil the the joint criteria of reducing its complexity compared to its antecedent model while maximizing its resemblance to its antecedent and keeping a similar performance score.

Remark 6.1. In this thesis, we refer to the original black-box model as an *antecedent* model and the simplified model as the *proxy* model. Furthermore, we qualify that all proxy models must be designed to globally approximate their respective antecedent models.

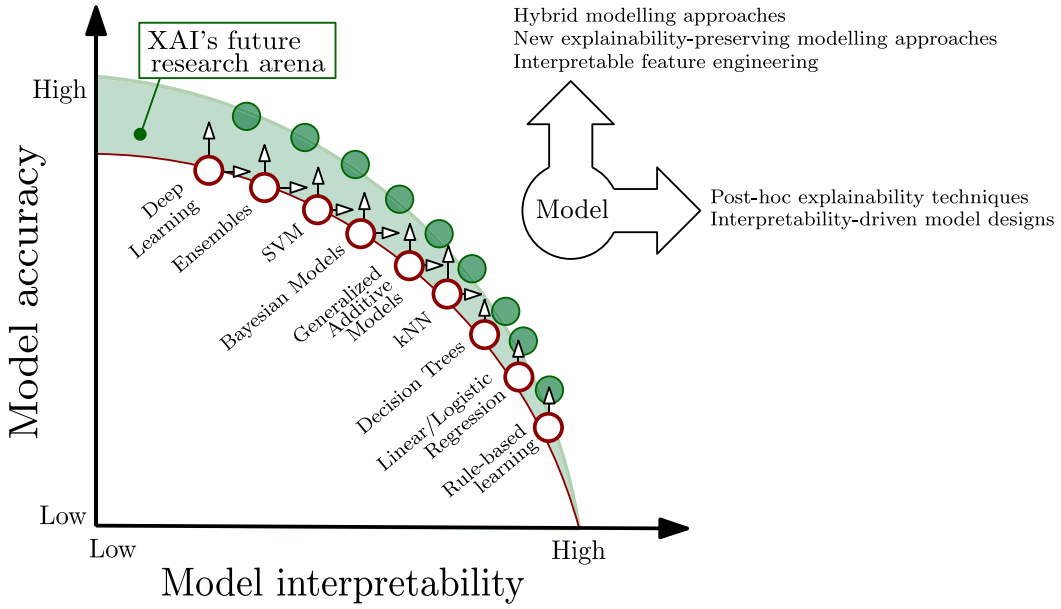

FIGURE 3: Schematic visualizing the performance-interpretability tradeoff; figure taken from Arrieta et al. (2020)

Remark 6.2. Bastani, Kim, and Bastani (2017) and Tan et al. (2018) are examples of studies that extract and distill simpler proxy models from complex antecedent models.

Through a survey of recent literature on explanations by simplification applied in the Natural Language Processing (NLP) field, we came across several prominent studies employing explanations by simplification to simplify antecedent black-box neural networks into proxy Finite-State Automata (FSAs) and/or Weighted Finite-State Automata (WFSAs) (Schwartz, Thomson, and Smith, 2018; Peng et al., 2018; Suresh et al., 2019; Wang and Niepert, 2019; Jiang et al., 2020). We expound more on WFSAs and Schwartz, Thomson, and Smith (2018) in Sections 2.3 and 2.4 respectively.

2.1.5 Performance-interpretability tradeoff

An interesting and insightful contribution of Arrieta et al. (2020, Section 5.1, Page 18) is the description of the performance-interpretability tradeoff; which they cautiously introduce as *"the matter of interpretability versus performance is one that repeats itself through time, but as any other big statement, has its surroundings filled with myths and misconceptions."* To address some of the aforementioned myths and misconceptions, Arrieta et al. (2020) first disprove the generic statement that more complex models are always more accurate by pointing to certain case studies to support this argument. In particular, they show that complex models are not necessarily more accurate in cases where the function to be modeled is not complex, data is well-structured and input features are available with high quality.

Next, Arrieta et al. (2020) provide the case where the aforementioned statement, that complex models perform better, tends to be true. According to the study, this is usually true when the function to be modeled is sufficiently complex and where the input data has high diversity or variance; and possibly contains significant noise. In

such cases, Arrieta et al. (2020) argue that the performance-interpretability tradeoff can be observed; as exemplified in Figure 3.

2.1.6 Explainability metrics

Towards the end of their study, Arrieta et al. (2020) note two major limitations of the current state of XAI research. Firstly, they observe the lack of a unified conceptualization of explainability between various studies. They furthermore acknowledge that their study, while limited, could provide a good starting point for other XAI studies. Secondly, Arrieta et al. (2020) note the lack of a unified metric that denotes how explainable any given model is. They further explain why developing such a metric has been a difficult process for many XAI studies; particularly because such a metric would entail incorporating psychological, sociological and cognitive elements to accommodate the goodness of fit of an explainability method to a certain target audience. Furthermore, incorporating such elements might involve significant amounts of subjectivity in the desired metric.

To reduce some of the aforementioned subjectivity involved, Miller (2019) and Arrieta et al. (2020) provide basic guidelines of what could constitute a good explanation based on human psychology, sociology and cognitive sciences. Firstly, they observe that explanations are better when *constrictive*; meaning an explanation is good if it not only explains why a model made decision X, but also why it made decision X over decision Y. Next, they suggest that good explanations should be able to communicate causal links over probabilities; which could be a challenge for black-box models which generally compute aggregate probabilities without necessarily considering causal links. Finally, they recommend that explanations are better when *selective*; meaning that a good explanation should be able to selectively provide the most important causal links instead of all possible causal links as these might be irrelevant or confusing to the target audience.

2.2 Straight-through estimator

Activation quantized neural networks refer to neural networks that contain layers which transmit piece-wise discrete signals. One of the main challenges in training such activation quantized neural networks is that their gradients tend to vanish almost everywhere because their derivatives typically default to zero. (Bengio, Léonard, and Courville, 2013; Courbariaux et al., 2016; Yin et al., 2019).

One significant workaround for this issue was proposed by Bengio, Léonard, and Courville (2013) through the introduction of the Straight-Through Estimator (STE). In the vanilla version of the STE, the STE neuron emits a signal of 1 when its input is strictly positive, and emits a zero signal in all other cases (Equation 1). The STE then uses a simple identity function to estimate the gradient during the backward pass (Equation 2). The vanilla STE is visualized through a schematic in Figure 4.

$$\mathrm{STE}(x) = \begin{cases} 1 & x > 0 \\ 0 & x \leq 0 \end{cases} \tag{1}$$

$$\mathrm{STE}'(x) = x \tag{2}$$

Aside from the vanilla STE, several other flavors of STEs have been proposed by studies such as Courbariaux et al. (2016) and Yin et al. (2019). In general, these studies have shown that activation quantized neural networks perform competitively

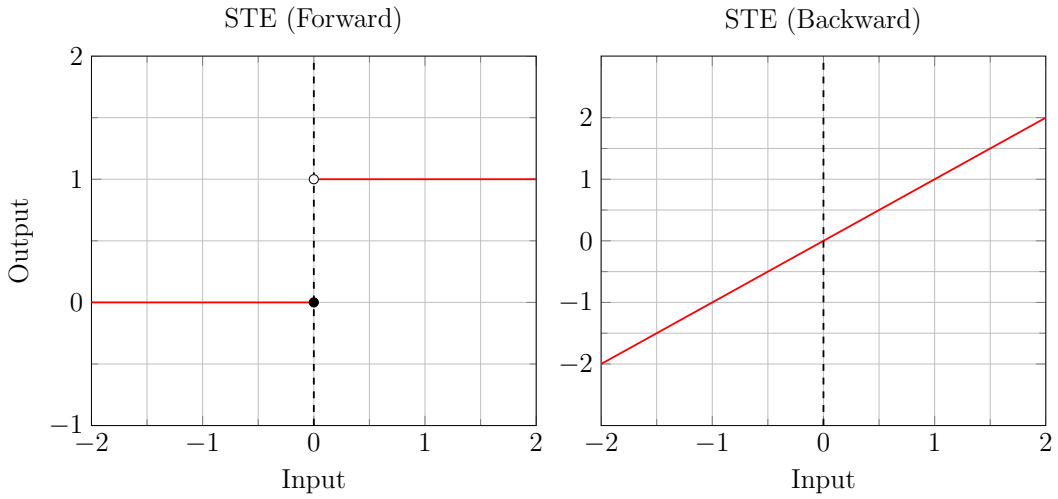

FIGURE 4: Schematic visualizing the vanilla STE's forward and backward passes

with their non-quantized counterparts; while providing performance-related benefits related to lower precision computing as well as enabling further research into threshold driven neural activation functions.

2.3 Weighted finite-state automaton

Definition 7 (Semiring; Kuich and Salomaa 1986)**.** A semiring is a set $\mathbb{K}$ along with two binary associative operations $\oplus$ (addition) and $\otimes$ (multiplication) and two identity elements: $\bar{0}$ for addition and $\bar{1}$ for multiplication. Semirings require that addition is commutative, multiplication distributes over addition, and that multiplication by $\bar{0}$ annihilates, i.e., $\bar{0} \otimes a = a \otimes \bar{0} = \bar{0}$.

Remark 7.1. Semirings follow the following generic notation: $\langle \mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1} \rangle$.

Remark 7.2. A simple and common semiring is the real or sum-product semiring: $\langle \mathbb{R}, +, \times, 0, 1 \rangle$. Two important semirings for this thesis are shown below.

Remark 7.3. **Max-sum** semiring: $\langle \mathbb{R} \cup \{-\infty\}, \max, +, -\infty, 0 \rangle$

Remark 7.4. **Max-product** semiring: $\langle \mathbb{R}_{>0} \cup \{-\infty\}, \max, \times, -\infty, 1 \rangle$

Definition 8 (Weighted finite-state automaton; Peng et al. 2018)**.** A weighted finite-state automaton over a semiring $\mathbb{K}$ is a 5-tuple $\mathcal{A} = \langle \Sigma, \mathcal{Q}, \mathcal{T}, \lambda, \rho \rangle$, with:

- a finite input alphabet Σ;
- a finite state set $\mathcal{Q}$;
- transition weights $\mathcal{T} : \mathcal{Q} \times \mathcal{Q} \times (\Sigma \cup \{\epsilon\}) \to \mathbb{K}$;
- initial weights $\lambda : \mathcal{Q} \to \mathbb{K}$;
- and final weights $\rho : \mathcal{Q} \to \mathbb{K}$.

Remark 8.1. Several studies refer to the transition weights $\mathcal{T}$ as the transition matrix, and the initial λ and final ρ weights as the initial and final weight vectors respectively (Schwartz, Thomson, and Smith, 2018; Jiang et al., 2020). As a result, we may use these terms interchangeably when referring to different studies.

Remark 8.2. $\epsilon \notin \Sigma$ refers to special ϵ-transitions that may be taken without consuming any input.

Remark 8.3. Self-loop transitions in $\mathcal{A}$ refer to special transitions which consume an input while staying at the same state.

Remark 8.4. Σ^* refers to the (possibly infinite) set of all strings over the alphabet Σ.

Definition 9 (Path score; Peng et al. 2018)**.** Let $\boldsymbol{\pi} = \langle \pi_1, \pi_2, \ldots, \pi_n \rangle$ be a sequence of adjacent transitions in $\mathcal{A}$, with each $\pi_i = \langle q_i, q_{i+1}, z_i \rangle \in \mathcal{Q} \times \mathcal{Q} \times (\Sigma \cup \{\epsilon\})$. The path $\boldsymbol{\pi}$ derives the ϵ-free string $\boldsymbol{x} = \langle x_1, x_2, \ldots, x_m \rangle \in \Sigma^*$; which is a substring of the ϵ-containing string $\boldsymbol{z} = \langle z_1, z_2, \ldots, z_n \rangle \in (\Sigma \cup \{\epsilon\})^*$. $\boldsymbol{\pi}$'s score in $\mathcal{A}$ is given by:

$$\mathcal{A}[\boldsymbol{\pi}] = \lambda(q_1) \otimes \left(\bigotimes_{i=1}^{n} \mathcal{T}(\pi_i) \right) \otimes \rho(q_{n+1}) \tag{3}$$

Definition 10 (String score; Peng et al. 2018)**.** Let $\Pi(\boldsymbol{x})$ denote the set of all paths in $\mathcal{A}$ that derive $\boldsymbol{x}$. Then the string score assigned by $\mathcal{A}$ to string $\boldsymbol{x}$ is given by:

$$\mathcal{A}[\![\boldsymbol{x}]\!] = \bigoplus_{\boldsymbol{\pi} \in \Pi(\boldsymbol{x})} \mathcal{A}[\boldsymbol{\pi}] \tag{4}$$

Remark 10.1. Since $\mathbb{K}$ is a semiring, $\mathcal{A}[\![\boldsymbol{x}]\!]$ can be efficiently computed using the Forward algorithm (Baum and Petrie, 1966). Its dynamic program is summarized below without ϵ-transitions for simplicity. $\Omega_i(q)$ gives the aggregate score of all paths that derive the substring $\langle x_1, x_2, \ldots, x_i \rangle$ and end in state q:

$$\Omega_0(q) = \lambda(q) \tag{5a}$$

$$\Omega_{i+1}(q) = \bigoplus_{q' \in \mathcal{Q}} \Omega_i(q') \otimes \mathcal{T}(q', q, x_i) \tag{5b}$$

$$\mathcal{A}[\![\boldsymbol{x}]\!] = \bigoplus_{q \in \mathcal{Q}} \Omega_n(q) \otimes \rho(q) \tag{5c}$$

Remark 10.2. The Forward algorithm can be generalized to any semiring (Eisner, 2002) and has a runtime of $O(|Q|^3 + |Q|^2|\boldsymbol{x}|)$ (Schwartz, Thomson, and Smith, 2018); notably with a linear runtime with respect to the length of the input string $\boldsymbol{x}$.

Remark 10.3. A special case of Forward is the Viterbi algorithm, where the addition $\oplus$ operator is constrained to the maximum function (Viterbi, 1967). Viterbi therefore returns the highest scoring path $\boldsymbol{\pi}$ that derives the input string $\boldsymbol{x}$.

2.4 Soft patterns

Schwartz, Thomson, and Smith (2018) present a novel hybridized RNN, CNN and WFSA-based neural architecture called **So**ft **Pa**tterns (SoPa). This architecture resembles a RNN because it processes text sequentially and can encode strings of arbitrary lengths. Similarly, the architecture contains a variable number of constrained linear-chain WFSAs with variable pattern lengths or window sizes; which resembles both one-layer CNNs and an ensemble of constrained linear-chain WFSAs. The combination of the aforementioned neural features allows SoPa to learn soft versions of traditional textual surface patterns.

In their study, Schwartz, Thomson, and Smith (2018) test the SoPa architecture on three separate sentiment classification tasks and compare the results with four baselines which included a bidirectional LSTM and a CNN. With their results, they show

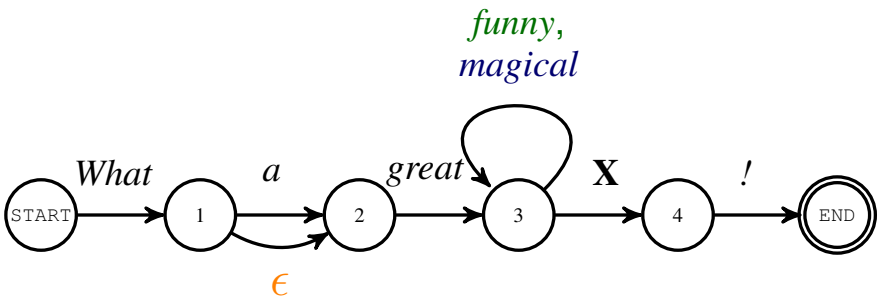

FIGURE 5: Schematic visualizing a linear-chain (weighted) finite-state automaton with one start and end state, ϵ-transitions, self-loops and a wildcard transition shown as **X**; note that wildcard transitions are not allowed in the SoPa model; figure taken from Schwartz, Thomson, and Smith (2018)

that SoPa performed on par or better than all baselines on all tasks. Additionally, they show that SoPa outperformed all baselines significantly in low-data settings. Finally, the authors present a simple method to explain the SoPa model using both local explanations and feature relevance.

2.4.1 Model specifications

WFSAs: In regards to the WFSAs used in SoPa, Schwartz, Thomson, and Smith (2018) only utilize the max-sum and max-product semirings; as well as ϵ-transitions and self-loops. Next, they employ constraints on the model's WFSAs in order to reduce the runtime of SoPa further. Of these, they firstly only allow one start and end state; which results in the initial λ and final ρ weights being reduced to static unitary values instead of state-dependent weights found in Definition 8. Next, they constrain ϵ-transitions to only occur at most once in each transition. Finally, they enforce a linear-chain WFSA structure which only allows consecutive-state transitions and therefore disallows transitions to previous states (Figure 5). This results in the transition matrix $\mathcal{T}$ being reduced to a sparse diagonal matrix. These constraints result in the runtime of the linear-chain WFSAs being reduced to $O(|Q||\boldsymbol{x}|)$ compared to the original runtime in Remark 10.2.

WFSAs as patterns: The relationship between regular expressions and (weighted) finite-state automata has been well-established in the theory of finite-state automata; with several algorithms detailing a deterministic conversion process between these entities (Thompson, 1968; Jiang et al., 2020). Since the relationship between regular expressions and textual patterns is clear, Schwartz, Thomson, and Smith (2018) refer to the linear-chain WFSAs in SoPa simply as *"patterns"*. For brevity, we adopt the same terminology.

String vs. document score: Definition 10 describes how a WFSA can be used to compute a string score. Since SoPa was intended to compute scores for entire documents and not just short strings, Schwartz, Thomson, and Smith (2018) propose computing the string score over all consecutive substrings in the document. The document score for a WFSA would represent an aggregated score over all consecutive substrings and would therefore also depend on the semiring used in the WFSAs.

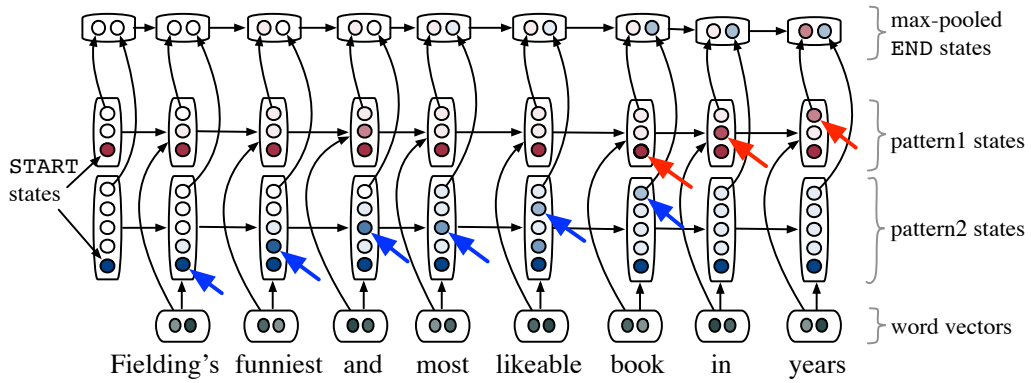

FIGURE 6: Schematic visualizing the SoPa model framework with two constituent WFSAs highlighted in blue and red; figure taken from Schwartz, Thomson, and Smith (2018)

In the case of max-based semirings using the Viterbi algorithm, the document score would reflect the highest scoring substring for a WFSA.

Computational graph: Figure 6 shows a sample computational graph for the SoPa model with two WFSAs highlighted in blue and red. The WFSAs compute consecutive string scores as they traverse the document. Given max-based semirings, the max-pooled scores accumulate in an output layer as shown on the top of Figure 6. After traversing the full document, the max-pooled scores are passed through a Multi-Layer Perceptron (MLP) which conducts the final classification to an output label. It is worth noting that without ϵ-transitions and self-loops, a linear-chain WFSA with $|\mathcal{Q}|$ states should always consume $|\mathcal{Q}| - 1$ tokens. However, by allowing the aforementioned special transitions; it is possible for strings of variable lengths to be consumed since an ϵ-transition can transition to the next state without consuming tokens while a self-loop can consume tokens without transitioning to the next state. This is indeed the case for Pattern 2 in Figure 6, when a self-loop is encountered in the transition from the token *"and"* to the token *"most"*.

Hyperparameters: Creating and training the SoPa model requires both commonly used and special hyperparameters. Commonly used hyperparameters include the learning rate, neuron dropout and word dropout. A special hyperparameter in the SoPa model is the pattern hyperparameter and it contains information on the number of patterns or WFSAs allowed, as well as the lengths of the patterns or the number of states in the WFSAs. This hyperparameter is encoded as a string with the following syntax: $\texttt{Length}_1\texttt{-Count}_1\texttt{\_}\ldots\texttt{\_Length}_n\texttt{-Count}_n$. An example of this hyperparameter could be `3-5_4-10_5-15`, which would signify 5 patterns of length 3, 10 patterns of length 4 and 15 patterns of length 5.

Explainability: In their study, Schwartz, Thomson, and Smith (2018) describe simple methods of *"interpreting"* the SoPa model. One method involves the usage of back-pointers during the Viterbi computation to determine the patterns and substrings in a document which contributed the highest pattern scores. Another method involves zeroing out the corresponding patterns or WFSAs via the occlusion sensitivity method to determine which pattern had the greatest impact on each classification decision. It is worth noting that the usage of *interpretation* in Schwartz, Thomson, and Smith (2018) is likely inconsistent with our definitions presented in Section

2.1. Given the common terminology misuse described in Section 2.1.3, it is more accurate to present the aforementioned *"interpretability"* method as an explainability method. We analyze the explainability-related features of SoPa in greater detail in Sections 2.4.2 and 2.4.3.

2.4.2 Transparency of SoPa

Since we expounded on XAI in Section 2.1 and made a case for viewing ML models from the lens of XAI, it would only make sense to extend the same standards to the SoPa model. Based on the arguments made by Arrieta et al. (2020), we can classify the SoPa model as a black-box model since it closely resembles RNNs and CNNs; and a strong case has already been made in their study regarding the black-box natures of both RNNs and CNNs. Naturally, this would imply that post-hoc explainability methods are required to explain the SoPa model.

2.4.3 Post-hoc explainability methods

Using the post-hoc explainability taxonomies described in Section 2.1.4, we can correspondingly classify the explainability methods presented in Schwartz, Thomson, and Smith (2018) using terminology consistent with XAI research. The first explainability method uses individual text samples to determine the highest scoring substrings in documents; as well as the patterns or WFSAs corresponding to them. Since this analysis is conducted at an individual document level and is never synthesized to a more global context, we would classify this under the local explanations explainability method. The next technique involves an occlusion or sensitivity analysis over all patterns and documents to determine which pattern had the greatest impact for each class. Since this involves systematic perturbation to determine the importance of pattern features, we would classify this method as a feature relevance explainability method.

Section 2.1.6 describes basic guidelines that could help elucidate what constitutes a good explanation; namely that a good explanation should be constrictive, provide causal links and be selective. We attempt to apply these guidelines with the aforementioned explainability techniques presented in SoPa. For the constrictive quality, it is likely that the explainability methods do not meet this criterion since they only highlight individual features that were important for SoPa, and do not necessarily go into detail regarding why these features superseded adjacent features. Next, the explainability methods likely do not fulfill the criterion of providing causal links; since they generally provide explanations by aggregating (ultimately) probabilistic quantities inside SoPa. Finally in contrast, the explainability methods likely pass the criterion of being selective since they only provide the most important features for an explanation.

Chapter 3

Methodologies

3.1 FMTOD data set

Schuster et al. (2019) originally released the Facebook Multilingual Task Oriented Dialog (FMTOD) data set to encourage research in cross-lingual transfer learning for Natural Language Understanding (NLU) tasks; specifically from from high-resource to low-resource languages. The authors released the FMTOD data set with English as the high-resource language with ∼43k annotated sentences, and Spanish and Thai as low-resource languages with a total of ∼14k sentences. Furthermore, they streamlined the data set on two key tasks; namely intent detection and textual slot filling. In this thesis, we focus solely on the English language intent detection task in the FMTOD data set. This intent detection task entails a multi-label sequence classification task with a total of 12 classes from alarm, reminder and weather-related domains.

3.1.1 Motivation

We chose to work with the FMTOD data set since it is both a recently released and well-studied data set (Schuster et al., 2019; Zhang et al., 2019; Zhang, Lyu, and Callison-Burch, 2020; Ren and Xue, 2020). We focus on the English language intent classification task since it is a relatively straightforward task which allows us to place a greater focus on performance and explainability. Furthermore, the English language subset entails the highest resources in the FMTOD data set. Finally, we find the FMTOD data set's intent detection classification especially attractive because it allows us to test the SoPa++ model on a multi-class NLU problem; which is significantly different from the focus on binary classification sentiment detection tasks in SoPa (Schwartz, Thomson, and Smith, 2018).

3.1.2 Preprocessing

We enumerate our preprocessing steps below:

1. Similar to Schwartz, Thomson, and Smith (2018), we convert all FMTOD text samples to a lowercased format. This assists in simplifying the data set further.

2. Next, we search through the pre-provided training, validation and test data partitions to remove duplicates within each partition.

3. Finally, we remove data duplicates which overlap between partitions. During this step, we do not remove any cross-partition duplicates from the test set in order to keep it as similar as possible to the original test partition. This comes into importance later when we compare performance evaluations on the test set with other studies.

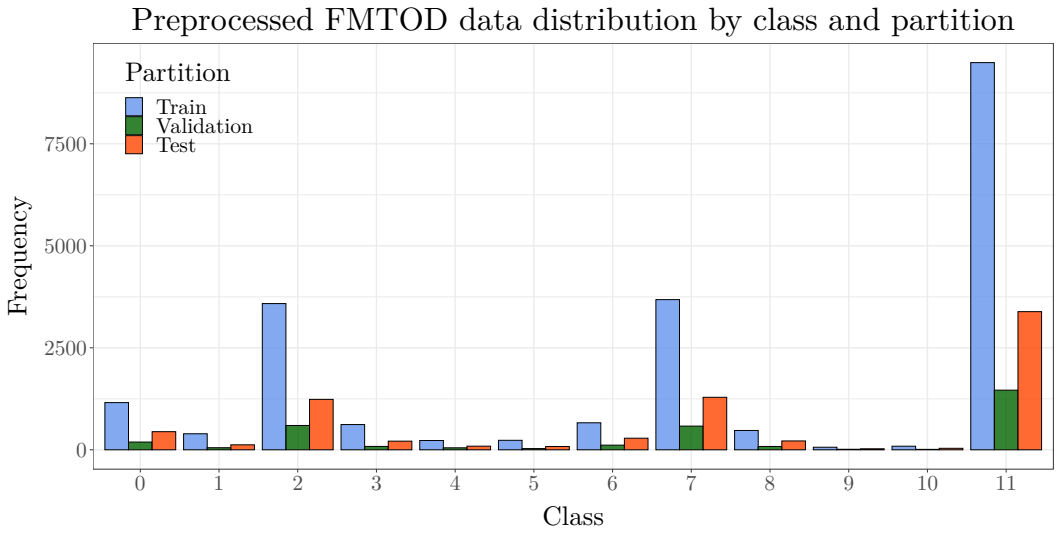

FIGURE 7: Data distribution of the preprocessed FMTOD data set grouped by classes and partitions

Class	Description	Train	Validation	Test	Combined
0	`alarm/cancel_alarm`	1157	190	444	1791
1	`alarm/modify_alarm`	393	51	122	566
2	`alarm/set_alarm`	3584	596	1236	5416
3	`alarm/show_alarms`	619	83	212	914
4	`alarm/snooze_alarm`	228	49	89	366
5	`alarm/time_left_on_alarm`	233	30	81	344
6	`reminder/cancel_reminder`	662	114	284	1060
7	`reminder/set_reminder`	3681	581	1287	5549
8	`reminder/show_reminders`	474	82	217	773
9	`weather/check_sunrise`	63	13	25	101
10	`weather/check_sunset`	88	11	37	136
11	`weather/find`	9490	1462	3386	14338
Σ	—	20672	3262	7420	31354

TABLE 1: Frequency of the preprocessed FMTOD data set classes grouped by partitions; Σ signifies the cumulative frequency in each data partition

Many of the duplicates observed were already present in the original FMTOD data set, with additional duplicates being created from the initial lowercasing step. After preprocessing, we obtain a lowercased variant of the FMTOD data set with strictly unique data partitions. In the next section, we describe the summary statistics of the preprocessed FMTOD data set.

3.1.3 Summary statistics

Figure 7 shows the summary statistics of the preprocessed FMTOD data set grouped by classes and data set partitions. Similarly, Table 1 shows the same summary statistics in a tabular form with explicit frequencies. Based on the summary statistics, we can observe that the preprocessed FMTOD data set is significantly imbalanced with

∼45% of samples falling into Class 11 alone. We take this observation into consideration in later sections and apply fixes to mitigate this data imbalance. In addition, we observe that input sentences in the preprocessed FMTOD data set are generally short; with a mean input token count of 7.7 and a standard deviation of 2.5 tokens.

3.1.4 Performance range

Several studies have optimized deep learning models on the FMTOD English language intent classification task using a variety of models from CNNs and BiLSTMs to BERT and XLM-RoBERTa (Schuster et al., 2019; Zhang et al., 2019; Zhang, Lyu, and Callison-Burch, 2020; Ren and Xue, 2020). The general accuracy range for the English language FMTOD intent classification task from the aforementioned studies is 96% to 99%.

Bibliography

Arrieta, Alejandro Barredo, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. (2020). "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI". In: *Information Fusion* 58, pp. 82–115.

Bastani, Osbert, Carolyn Kim, and Hamsa Bastani (2017). "Interpretability via model extraction". In: *arXiv preprint arXiv:1706.09773*.

Baum, Leonard E and Ted Petrie (1966). "Statistical inference for probabilistic functions of finite state Markov chains". In: *The annals of mathematical statistics* 37.6, pp. 1554–1563.

Bengio, Yoshua, Nicholas Léonard, and Aaron Courville (2013). "Estimating or propagating gradients through stochastic neurons for conditional computation". In: *arXiv preprint arXiv:1308.3432*.

Courbariaux, Matthieu, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio (2016). "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1". In: *arXiv preprint arXiv:1602.02830*.

Danilevsky, Marina, Kun Qian, Ranit Aharonov, Yannis Katsis, Ban Kawas, and Prithviraj Sen (2020). "A survey of the state of explainable AI for natural language processing". In: *arXiv preprint arXiv:2010.00711*.

Doran, Derek, Sarah Schulz, and Tarek R. Besold (2017). "What Does Explainable AI Really Mean? A New Conceptualization of Perspectives". In: *CoRR* abs/1710.00794. arXiv: 1710.00794. URL: http://arxiv.org/abs/1710.00794.

Eisner, Jason (2002). "Parameter estimation for probabilistic finite-state transducers". In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 1–8.

Jiang, Chengyue, Yinggong Zhao, Shanbo Chu, Libin Shen, and Kewei Tu (2020). "Cold-start and Interpretability: Turning Regular Expressions into Trainable Recurrent Neural Networks". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 3193–3207.

Konig, Rikard, Ulf Johansson, and Lars Niklasson (2008). "G-REX: A versatile framework for evolutionary data mining". In: *2008 IEEE International Conference on Data Mining Workshops*. IEEE, pp. 971–974.

Kuich, Werner and Arto Salomaa (1986). "Linear Algebra". In: *Semirings, automata, languages*. Springer, pp. 5–103.

Lundberg, Scott and Su-In Lee (2017). "A unified approach to interpreting model predictions". In: *arXiv preprint arXiv:1705.07874*.

Miller, Tim (2019). "Explanation in artificial intelligence: Insights from the social sciences". In: *Artificial Intelligence* 267, pp. 1–38. ISSN: 0004-3702. DOI: https://doi.org/10.1016/j.artint.2018.07.007. URL: https://www.sciencedirect.com/science/article/pii/S0004370218305988.

Peng, Hao, Roy Schwartz, Sam Thomson, and Noah A. Smith (2018). "Rational Recurrences". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural*

Language Processing. Brussels, Belgium: Association for Computational Linguistics, pp. 1203–1214. DOI: 10.18653/v1/D18-1152. URL: https://www.aclweb.org/anthology/D18-1152.

Ren, Fuji and Siyuan Xue (2020). "Intention detection based on siamese neural network with triplet loss". In: *IEEE Access* 8, pp. 82242–82254.

Ribeiro, Marco Tulio, Sameer Singh, and Carlos Guestrin (2016). ""Why Should I Trust You?": Explaining the Predictions of Any Classifier". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pp. 1135–1144.

Sanh, Victor, Lysandre Debut, Julien Chaumond, and Thomas Wolf (2019). "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter". In: *arXiv preprint arXiv:1910.01108*.

Schuster, Sebastian, Sonal Gupta, Rushin Shah, and Mike Lewis (June 2019). "Cross-lingual Transfer Learning for Multilingual Task Oriented Dialog". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 3795–3805. DOI: 10.18653/v1/N19-1380. URL: https://www.aclweb.org/anthology/N19-1380.

Schwartz, Roy, Sam Thomson, and Noah A. Smith (July 2018). "Bridging CNNs, RNNs, and Weighted Finite-State Machines". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, pp. 295–305. DOI: 10.18653/v1/P18-1028. URL: https://www.aclweb.org/anthology/P18-1028.

Suresh, Ananda Theertha, Brian Roark, Michael Riley, and Vlad Schogol (2019). "Approximating probabilistic models as weighted finite automata". In: *CoRR* abs/1905.08701. arXiv: 1905.08701. URL: http://arxiv.org/abs/1905.08701.

Tan, Sarah, Rich Caruana, Giles Hooker, and Yin Lou (2018). "Distill-and-compare: Auditing black-box models using transparent model distillation". In: *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 303–310.

Thompson, Ken (1968). "Programming techniques: Regular expression search algorithm". In: *Communications of the ACM* 11.6, pp. 419–422.

Townsend, Joseph, Thomas Chaton, and João M Monteiro (2019). "Extracting relational explanations from deep neural networks: A survey from a neural-symbolic perspective". In: *IEEE transactions on neural networks and learning systems* 31.9, pp. 3456–3470.

Viterbi, Andrew (1967). "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm". In: *IEEE transactions on Information Theory* 13.2, pp. 260–269.

Wang, Cheng and Mathias Niepert (2019). "State-Regularized Recurrent Neural Networks". In: ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, pp. 6596–6606. URL: http://proceedings.mlr.press/v97/wang19j.html.

Yin, Penghang, Jiancheng Lyu, Shuai Zhang, Stanley Osher, Yingyong Qi, and Jack Xin (2019). "Understanding straight-through estimator in training activation quantized neural nets". In: *arXiv preprint arXiv:1903.05662*.

Zeiler, Matthew D and Rob Fergus (2014). "Visualizing and understanding convolutional networks". In: *European conference on computer vision*. Springer, pp. 818–833.

Zhang, Li, Qing Lyu, and Chris Callison-Burch (2020). "Intent Detection with WikiHow". In: *arXiv preprint arXiv:2009.05781*.

Zhang, Zhichang, Zhenwen Zhang, Haoyuan Chen, and Zhiman Zhang (2019). "A joint learning framework with bert for spoken language understanding". In: *IEEE Access* 7, pp. 168849–168858.