COLUMBIA UNIVERSITY
Data Science Institute

ENGI E4800: Data Science Capstone and Ethics

Final Report

Written by:[1]

Timothy Hao Huang

Jay Zern Ng

Yuki Nishimura

Jonathan Irvin Santoso

Kevin Christian Wibisono

Supervised by:

Professor Yuval Marton (Bloomberg CTO Office)

Professor Asad Basheer Sayeed (University of Gothenburg)

Professor Smaranda Muresan (Columbia University)

Fu Foundation School of Engineering and Applied Science

[1] In alphabetical order

Contents

1 Introduction

Despite continuous advancements in the field of artificial intelligence (AI) and natural language processing (NLP), machines are still not intelligent enough to "understand" events from text, let alone to emulate human level performance. The vast amount of text data or corpus available in the world further engenders the need to develop tools for this purpose. These tools should, for example, be able to inform the decision to buy or sell stocks based on the sentence "company X is teaming up with company Y on company Z bid".

In the understanding of events, the relationship between the components of an event is an important factor which naturally relates to a subfield in NLP known as *thematic fit*: given a verb v and an entity x,evaluate how well v fits x in role r. For example, given verb-entity tuples (*cut*, *knife*), and (*cut*, *bowl*), and the role *instrument*, the former tuple will have better thematic fit, since *knife* is more likely to be used to *cut* an object. Tilk et al. (2016) simulated thematic fit via *selectional preferences*, i.e. generating a $(|V|-1)$-simplex distribution of possible words to take on a specific role (known as *role-fillers*). In particular, given a set of input (word, role) pairs and a target role, the model (called *neural network non-incremental role-filler* or **NNRF**) predicts the correct role-filler for the target role.

Tilk et al.'s (2016) model architecture can be summarized as follows: for each input word, a role-specific word embedding is calculated from the factored embedding tensor, which are then added together and fed to a nonlinear activation layer; this layer is finally fed into a softmax output layer through the role-specific factored classifier tensor. Despite its state-of-the-art performance on well-known thematic fit tasks, the **NNRF** model suffered from a major limitation. This model does not properly utilize thematic role input, as evidenced by the sentences *apple eats boy* and *boy eats apple* having similar representations.

This limitation is the key motivation behind Hong et al.'s (2018) multi-task learning approach, which added a secondary role prediction task to the model. They invented a new state-of-the-art architecture called **ResRoFA-MT**, which has been shown to result in superior performance in thematic fit and event similarity tasks. This model is derived from **NNRF** by (1) adding a secondary task of predicting a target role given a target word; (2) introducing residual blocks; and (3) calculating the event representation embedding through weighted average of role-filler embeddings.

Hong et al.'s (2018) model also performs well in psycholinguistically-motivated as well as application-based semantic evaluation tasks. One example is *thematic fit correlation*, which uses Spearman's correlation to correlate the softmax outputs from role-filler predictions with ratings from participant-based questionnaires from McRae et al. (2005) [1,444 verb-agent or verb-patient ratings], Pado (2007) [414 verb-agent or verb-patient ratings], Ferretti (2001) [274 verb-location or verb-instrument ratings] and Greenberg et al. (2015) [720 verb-patient ratings]. Another one is *semantic role classification*, which involves how well the newly added secondary component of the multi-task model is able to predict semantic roles on the CoNLL-2005 shared task (Carreras and Màrquez, 2005).

Our work focuses on experimenting with new architectures which aim to improve the performance of the current state-of-the-art **ResRoFA-MT** model (subsequently referred to as the *baseline*) mainly in terms of thematic fit evaluation, but will also assess target role and target word prediction accuracy. As directions of potential improvements to the model, 6 different parts of the model are focused on: input style, target word input timing, aggregation, non-sequential input, activation function, and target objective. Experiments are conducted on a part-by-part basis to make comparisons with the baseline simple and clear, all of which will be explained in detail in Section 4.

2 Background and Literature Review

2.1 Baseline Model Architecture

The current baseline model used for this project is the state-of-the-art **ResRoFA-MT** (Hong et al., 2018), which is a residual network based architecture with multi-task capabilities to predict both the target role and word given a set of input roles and words. Figure 1 below provides a visual of the baseline architecture.

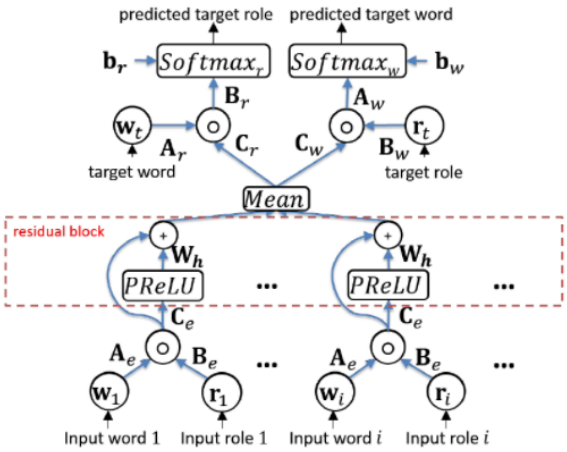

Figure 1: Baseline ResRoFA-MT architecture

This subsection explains the mathematics behind the **ResRoFA-MT** architecture based on Hong et al.'s (2018) supplemental document. Let $\mathbf{T} \in \mathbb{R}^{|V|\times|R|\times d}$ be a role-specific embedding tensor, where $|V|$ denotes the vocabulary size, $|R|$ the number of possible roles and d the embedding dimension. In order to reduce the number of parameters, the tensor rank decomposition is introduced. This decomposition transforms $\mathbf{T}$ into $\sum_{m=1}^{k} \mathbf{a}_m \otimes \mathbf{b}_m \otimes \mathbf{c}_m$, where $\mathbf{a}_m \in \mathbb{R}^{|V|}$, $\mathbf{b}_m \in \mathbb{R}^{|R|}$, $\mathbf{c}_m \in \mathbb{R}^d$, and $\otimes$ denotes the outer product. Here, k is defined to be the tensor rank of $\mathbf{T}$.

For ease of explanation, the $\mathbf{a}_m$'s are concatenated by column to form $\mathbf{A}_e \in \mathbb{R}^{|V|\times k}$, the $\mathbf{b}_m$'s by column to form $\mathbf{B}_e \in \mathbb{R}^{|R|\times k}$, and the $\mathbf{c}_m$'s by row to form $\mathbf{C}_e \in \mathbb{R}^{k\times d}$. The event-participant embedding vector $\mathbf{T}_{(ij)} \in \mathbb{R}^d$ (for any word i and role j) can now be calculated as $(\mathbf{A}_{e(i)} \circ \mathbf{B}_{e(j)})\mathbf{C}$, where $\circ$ denotes the Hadamard product. Similarly, it can be easily shown that $\mathbf{p}_l = \mathbf{T}_{(ij)} = (\mathbf{w}_i\mathbf{A}_e \circ \mathbf{r}_j\mathbf{B}_e)\mathbf{C}_e$, where w_i and r_j denote the one-hot encoding of word i and role j, respectively. The event embedding can be simply expressed

as $\mathbf{e} = \frac{1}{|C|}\sum_{l \in C} PReLU(\mathbf{p}_l)$, where $|C|$ denotes the number of (input word, input role) pairs (assuming no residual block at this moment).

Next, the weight matrices $\mathbf{W}_w$ and $\mathbf{W}_r$ are defined for each target word a_x and target role b_y. Stacking $\mathbf{W}_w$ for all words and $\mathbf{W}_r$ for all roles result in weight tensors $\mathbf{T}^{(w)} \in \mathbb{R}^{d \times |R| \times |V|}$ and $\mathbf{T}^{(r)} \in \mathbb{R}^{d \times |V| \times |R|}$. Applying the same tensor rank decomposition technique as above and concatenating the resulting vectors by row or column to simplify expressions, it is obtained that $\mathbf{W}_w = \mathbf{C}_w \text{diag}(\mathbf{b}_y)\mathbf{A}_w$ and $\mathbf{W}_r = \mathbf{C}_r \text{diag}(\mathbf{a}_x)\mathbf{B}_r$, where $\mathbf{C}_w \in \mathbb{R}^{d \times k^{(w)}}$, $\mathbf{C}_r \in \mathbb{R}^{d \times k^{(r)}}$, $\mathbf{A}_w \in \mathbb{R}^{k^{(w)} \times |V|}$, and $\mathbf{A}_r \in \mathbb{R}^{k^{(r)} \times |V|}$. Here, $k^{(r)}$ and $k^{(w)}$ denote the tensor rank of $\mathbf{T}^{(r)}$ and $\mathbf{T}^{(w)}$, respectively. Upon defining the target role embedding matrices $\mathbf{B}_w \in \mathbb{R}^{k^{(w)} \times |R|}$ and target word embedding matrix $\mathbf{A}_r \in \mathbb{R}^{k^{(w)} \times |R|}$, the weight matrices can be rewritten as $\mathbf{W}_w = \mathbf{C}_w \text{diag}(\mathbf{r}_y \mathbf{B}_w)\mathbf{A}_w$ and $\mathbf{W}_r = \mathbf{C}_r \text{diag}(\mathbf{w}_x \mathbf{A}_r)\mathbf{B}_r$. Now, given a target word a_x and target role b_y, each of $\mathbf{W}_w$ and $\mathbf{W}_r$ is fed into a dense layer with a softmax activation function in order to output the probability vectors of roles and words.

With residual blocks, the $\mathbf{T}_{ij}$ is passed to a PReLU layer, i.e. $\mathbf{h}_l = PReLU(\mathbf{r}_l \mathbf{C}_e)$, where $\mathbf{r}_l = \mathbf{w}_i \mathbf{A}_e \circ \mathbf{r}_j \mathbf{B}_e$. $\mathbf{h}_l$ and $\mathbf{r}_l$ then form an event embedding as governed by the following equation: $\mathbf{e} = \frac{1}{|C|}\sum_{l \in C}(\mathbf{h}_l \mathbf{W}_h + \mathbf{r}_l)$. Since this is a multi-task architecture, the loss function must be a combination of the losses of both tasks. In **ResRoFA-MT**, the loss function is simply defined as summing (or equivalently averaging) the cross-entropy loss obtained from both word and role predictions tasks.

2.2 Thematic Fit Estimation

Prior to Tilk et al.'s (2016) work, models pertaining to thematic fit have been developed using a distributional memory (DM) framework, which lacks the ability to optimize the distributional space in a principled way. Departing from Erk et al. (2010)'s work on syntax-based distributional semantic models (DSMs), Baroni and Lenci (2010) proposed a DM framework based on prototype vectors by averaging dependency-based vectors of typical roles. Sayeed and Demberg (2014) then proposed a DSM-like structure which used a neural network-based SENNA semantic role labeler to define the feature spaces. A year after, Greenberg et al. (2015) developed a TypeDM or role-based model, which measured the effects of verb polysemy on thematic fit.

Tilk et al.'s (2016) work was the first to incorporate a neural network-based architecture for thematic fit. The rationale behind their architectural choice was the capability of a neural network to optimize the distributional representation for the task, an aspect which hindered the previous models from achieving superior performance on thematic fit. Two years afterwards, Hong et al. (2018) proposed an improvement to Tilk et al.'s (2016) non-incremental architecture by introducing a secondary task, allowing the model to simultaneously predict event participants and classify semantic roles.

2.3 Evaluation Metrics

The models developed were evaluated based on one intrinsic task and one extrinsic task: (1) prediction accuracy (intrinsic) (2) thematic fit correlation (extrinsic).

2.3.1 Prediction Accuracy

Due to the limited data available for evaluating natural language understanding, the models were trained to predict next word given role labels, and next role given the word labels. This evaluation technique resulted in the intrinsic evaluation metric of test/validation **word and role accuracies** which in theory has an indirect positive correlation with thematic fit scores.

2.3.2 Thematic Fit

For thematic fit correlation, two different datasets are utilized for evaluation. All the human judgement ratings presented in the datasets range from 1 (least common) to 7 (most common).

- **Pado07** (Pado, 2007): 414 balanced pair of 18 verbs and 12 nouns extracted from WSJ corpus. Format: agent/patient/role/ratings (e.g. advise, doctor, ARG0, 6.8)
- **McRae05** (McRae et al., 2005): 1444 unbalanced pair ratings set. Format: agent/ patient/role/ratings (e.g. advise, doctor, ARG0, 6.8)

The model was tasked to predict scores in order to obtain the Spearman's correlation between predicted scores and human judgment scores.

2.4 Potential Issues of Baseline

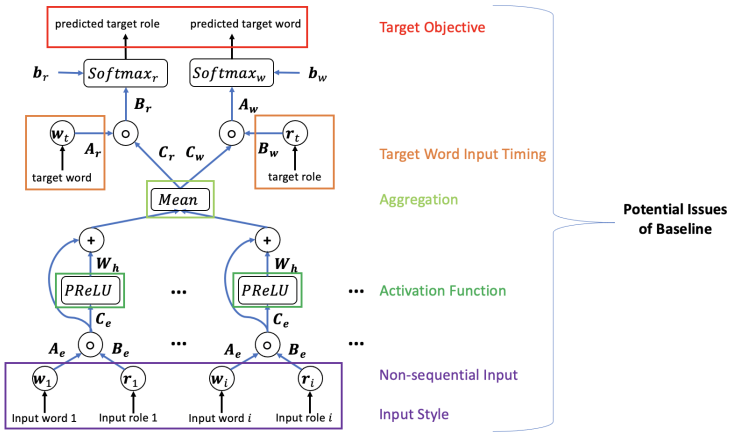

Figure 2: Potential Issues of Baseline Model

In the baseline model ResRoFA-MT, six potential issues were identified and are presented in Figure 2: input style, non-sequential input, activation function, aggregation, target word input timing, target objective. Each of these issues can be investigated and tested by a new architecture focusing on one change at a time. Input style refers to adding one-hot encoding to each input word and role in the input layer. The baseline model also does not take word order into account, so modifying the model to support sequential information would address the non-sequential input issue. Modifying the activation function used in the baseline model could also optimize model performance. Furthermore, the mean aggregation used in the baseline model could be modified or replaced to allow the model to provide a better summarization of word-role representations. Target word input timing refers to when the target word and target role is introduced in the model.

These could be introduced earlier in the model to provide more information about the target and improve event representation. Finally, since the baseline model utilizes multi-task learning, modifying the target objective to give more weight to a specific task could affect model outcomes.

Before diving into further detail about each of the new architectures created to address these issues, we will first give an overview of the data and experimental setup.

3 Preparation

3.1 Data Overview

The dataset used for the project is the Rollenwechsel-English (RW-eng) corpus, which is a large corpus of automatically labelled semantic frames extracted from ukWaC and BNC using Propbank roles. Currently, there are two available versions of the dataset: V1 (Sayeed et al., 2018) and V2 (Sayeed and Marton, Submitted).

For V1, the dataset is preprocessed by initially passing through the Malt-Parser and tagged with part-of-speech (POS) tags through syntactic parse trees. Afterwards, the dataset is parsed using SENNA for semantic role labelling. SENNA (Collobert et al. 2011), which does not rely directly on the syntactic parse of the sentence, outputs predicate words and spans of text connected to the predicate with Propbank style roles. If the spans of text connected to the predicates are not singleton, the spans of text will further be processed using three different heuristics (MALT, MALT-SPAN, LINEAR) to identify head words. If none of these three parsers can detect headwords, the span will be labelled as FAILED and treated as a single full constituent. The output format of the pipeline is an XML file that contains multiple predicate tags per sentence, one governor tag per predicate, and multiple dep tags per predicate. Moreover, in each of the tags, lemmatized texts are provided combined with words which are annotated in the format word/POS/N where N is the position of the word relative to the first word in the sentence starting at 1. Figure 3 shows the schematic of the XML files output for the V1 dataset.

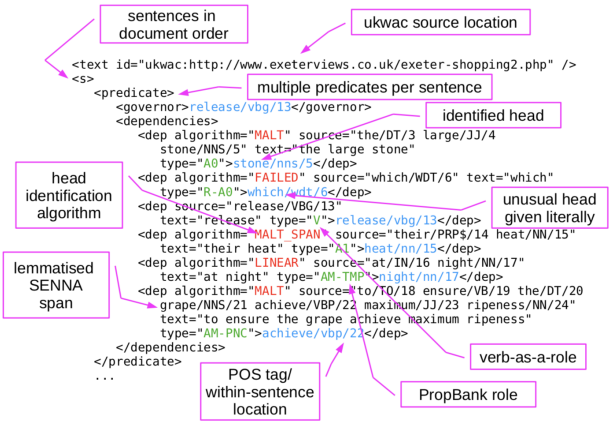

Figure 3: Excerpt of a single annotated predicate from the RW-eng corpus (Sayeed et al., 2018)

For V2, the output format of the dataset is like that of V1. However, SpaCy (spacy.io) dependency parser is used in place of the Malt-Parser, and SENNA is replaced with He-

SRL (He et al., 2017) to obtain a more accurate prediction for both predicates and their dependencies. In addition, frame tags were added to include the output of He-SRL with the use of LSGN tagger.

Overall, the RW-eng corpus contains more than 78M sentences, 2.3M documents, 210M predicates and 704M role-fillers. The outputs of SENNA/He-SRL were further preprocessed to create inputs for modeling. The top 50K most common words plus 1 out of vocabulary (OOV) token combined with seven unique role tags (PRD [predicate], ARG0 [agent-like], ARG1 [patient-like], ARGM-MNR [manner], ARGM-LOC [location], ARGM-TMP [temporal], OTHERS) were used for mapping. Due to the large data size, several training sets utilizing 0.1% (approx. 150k rows), 1% (approx. 1.6M rows), and 10% (approx. 17M rows) of the data was created while the development and test sets were predetermined to contain 0.4% of the data (approx. 800K rows each). Each of these files were stored in disk for ease of access during modeling.

The train/dev/test files contains a dictionary of length seven per line with keys referring to the role ids and values referring to word ids. Each of the known word ids in each input line is removed one at a time to generate the input data and labels which are later sent to the model via Python generator expressions. Figure 4 describes how raw files are turned into input and label data used for modeling.

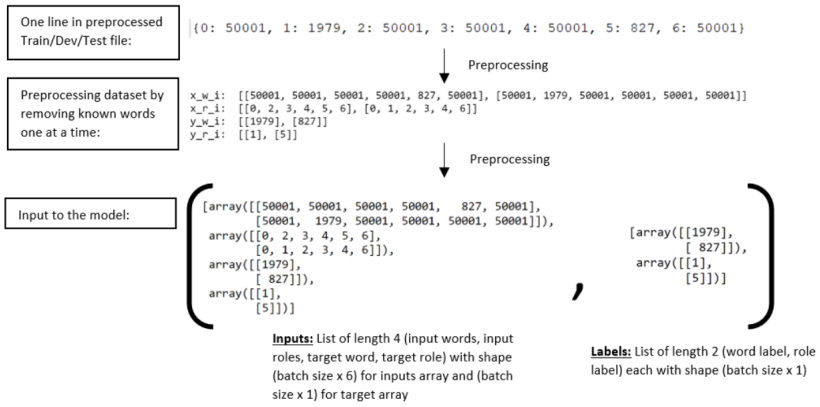

Figure 4: Transformation from preprocessed files to model input

3.2 Method

3.2.1 Environment Setup

Our code is version controlled through a Github repository and primarily developed in two environments: Virtual Machine (VM), and Google Colab. For VM development, Google Cloud was used to create Deep Learning VM with 26GB RAM and T4 GPU running Python 3.7 and Tensorflow 2.3.0 environment at a cost of $0.4 per hour. Google Colab was primarily used to prototype and test new code which involved migrating from Github into Google Drive. In both environments, the only dependencies missing is nltk data which can be downloaded by running `import nltk; nltk.download("all")`.

In order to validate the original model implementation and conversion, we also used the Docker image provided in Hong et al.'s (2018) with Python 2.7 and Keras 2.0 environment to run the original implementation with Theano backend.

3.2.2 Conversion of Code to Python 3.7 and Tensorflow 2.3.0

The original code was written in Python 2.7 with Keras 2.0 and Theano, which is currently unsupported apart from the old Docker image. Prior to implementing the new models in order to draw comparisons with the current baseline models, the decision was made to convert the legacy code into Python 3.7 and Tensorflow 2.3.0.

Several key changes were made apart from the syntax conversion from 2.7 to 3.7; These were creating a custom accuracy metric for Tensorflow to evaluate the model, modifying file I/O functions in *batcher.py* to prevent encoding errors, and finally changing the optimizer from Adagrad to Adam. In conjunction with the latter, the learning rate was tuned from 0.1 to 0.01 which helped models to avoid being stuck at a local minima during the training period.

0.1% of data	Python 2.7 + Keras 2.0 (Prof Yuval's numbers)		Python 2.7 + Keras 2.0 (Theano backend)		Python 3.7 + Tensorflow 2.3.0	
Data Version	v1	v2	v1	v2	v1	v2
Optimizer + Learning Rate	Adagrad 0.1	Adagrad 0.1	Adagrad 0.1	N/A	Adam 0.01	Adam 0.01
Val/Test Role Accuracy	~88%	~90-91%	89.60%	N/A	90.30%	91.70%
Val/Test Word Accuracy	~4%	~8-10%	4.80%	N/A	4.20%	9.20%
Pado-All	~23-39	~29-35	22	N/A	24.6	29.4
McRae-All	~18-20	~18-20	17.2	N/A	18.6	17.5

Table 1: Summary of ResRoFA-MT performance on 0.1% data using various implementations

The original and converted code was left to train on the same 0.1% dataset to produce results that was within the margin of error as shown in Table 1 to validate the code conversion process

3.3 Sequential Preprocessing

3.3.1 Input Data

The sequential ordering of words were ignored in both V1 and V2 datasets. As we hypothesize that the ordering of words is salient for the task of thematic fit estimation, ordering information is included into both the V1 and V2 datasets. Instead of keeping the ordering of the role-ids constant, the role-ids were ordered in the appearance order of the corresponding words, which were obtained by tracking N from the given word/POS/N format in the datasets, where N is the position of the word relative to the first word, to incorporate sequential information into the datasets. Python 3.7+ offered ordered dictionaries as a default feature of dictionaries, and was utilized to keep the data format consistent while taking into account the sequential orderings. In the case of out of vocabulary (OOV) tokens with word-id 50001, the roles were ordered in increasing order within the ordered dictionary, after the roles corresponding to existing vocabulary. Figure 5 is an example of the sequentially preprocessed data (same format for both V1 and V2 datasets). Note that through this preprocessing, there were cases where the vocabulary dictionary word-ids became inconsistent between non-sequential and sequential data (Ex. word-id of the word "banker" in non-sequential data was 1310, whereas it was 1284 in sequential data).

Non-Sequential	{0: 50001, 1: 883, 2: 50001, 3: 50001, 4: 50001 , 5: 142, 6: 50001}
Sequential	{5: 142, 1: 883, 0: 50001, 2: 50001, 3: 50001, 4: 50001, 6: 50001}

Figure 5: A single dictionary sample of a row in the preprocessed files

3.3.2 Evaluation Metrics

As a result of the difference in preprocessing input pipeline for sequential models, further modification to the evaluation files was needed to align the format with the preprocessed inputs. In the original evaluation files, the evaluation data was passed on to the model based on the input ordering specified in `description.txt` file in the data folder. In the modified evaluation files, word ordering is taken into consideration where words and roles that occur first are placed at the start of the input array. Then, word input arrays are post-padded with the Missing word token (50001) until length 6, and role input arrays are post-padded with other roles not present in the input/label in ascending order. The role padding ensures that all 7 roles are present in either the input or label array.

For example, in Pado evaluation, the line "advise banker ARG0 6.0" was converted to input word [50001, 50001, 50001, 1310, 50001, 50001] by the original evaluation files because of the fixed ordering of the roles specified in the description file as [4,2,1,5,3,6]. As a result, known word ids (non-50001) could be found anywhere throughout the sequence. However, in the modified process, the same input would be preprocessed to [1310, 50001, 50001, 50001, 50001, 50001] and [5, 1, 2, 3, 4, 6] such that known word ids are placed in front of the padding tokens.

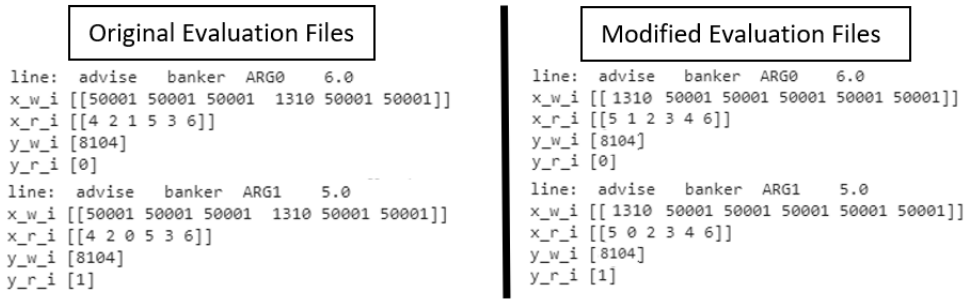

Figure 6: Input to evaluation files for word probability prediction

Figure 6 shows the difference between the original evaluation files and modified evaluation files for Pado and McRae thematic fit evaluation. These modifications were performed across all five evaluation files (Pado, McRae, Bicknell, Greenberg, GS) such that the input arrays could be passed on to the models to obtain the next word probabilities given the target word.

4 Proposed Model Architectures

For each of the aforementioned issues of the ResRoFA-MT architecture, modifications which could potential improve the performance of thematic fit were made on the architecture. To better understand the reasoning behind improvements or deterioration, we ensured the architecture was modified on a part-by-part basis.

4.1 Modification of Input Style

To address the input style of the baseline model, a new architecture was proposed based on a well known wide-and-deep learning architecture (Cheng et al., 2016), which has been shown to perform well on recommendation tasks. The model's main feature is the combination of sparse and dense information. Our ResRoFWD-MT model implementation, shown in Figure 14, modifies the baseline by adding a one-hot encoding of each input word and role and combines it with the input word-role embedding before being passed into the PReLU layer.

4.2 Modification of Target Word Input Timing

The baseline model introduces the target word and the target role after the dense layer. An attempt at modifying the target word input timing would be introducing the target word and role earlier in the model. The ResRoFBeg-MT model, as shown in Figure 15, achieves this by introducing the target word and role in the dense layer. This design would potentially improve model performance as the event representation will have information about the target. This modification reduces the tensor factorization from 2 to 1, but introduces two task-specific dense layers.

4.3 Modification of Aggregation

Recall that the baseline model uses a mean aggregation layer to generate the event representation. We attempted to substitute this aggregation layer using a dense layer (with the same number of neurons) as shown in Figure 16. We call this model ResRoFDense. The rationale behind this is to allow the neural networks to automatically find a functional form which best summarizes the (input word, input role) representations in the form of an event embedding. Some simple modifications of this model involved adjusting the number of dense layers and neurons in each layer. For an apples-to-apples comparison with the baseline model, we chose a single dense layer shown in Figure 16.

4.4 Modification of Non-Sequential Input

4.4.1 Applying RNN, LSTM, BiLSTM

One possible attempt at improving ResRoFA-MT would be to take the ordering of the input word-role pairs into account. This implies that we need to design a way to combine the individual word-role embeddings (i.e. the parametric ReLU outputs) to form an aggregated event embedding in a way that incorporates event participants' ordering. Not only that, this attempt induces the need for modified input preprocessing and evaluation scripts which ensure correct ordering of the event participants, which has been explained in Section 3.3.

The architectures we developed are based on recurrent neural networks (RNNs), well-known for their ability to connect previous information to the present task (Olah, 2015). Figure 17 illustrates an unrolled RNN layer.

Starting with a sequence of inputs $(x_1, x_2, \cdots, x_T)$, an RNN produces sequences of hidden states $(a_1, a_2, \cdots, a_T)$ and output states $(h_1, h_2, \cdots, h_T)$ according to the equations $a_t = \tanh(b + Wa_{t-1} + Ux_t)$ and $h_t = c + Va_t$, where U, V, W, b, c are learnable

parameters. The output state corresponding to the last time step (i.e. h_T) should then be able to "summarize" information provided by the input sequence.

Long short term memory networks (LSTMs) (Hochreiter and Schmidhuber, 1997) are an extension of RNNs designed to learn long distance dependencies, i.e. remembering information for long periods of time. This property is achieved by adding gates to add, remove or forget information over time. Mathematically, we can express this through a "forget gate layer" f_t, "input gate layer" i_t, "output layer" o_t and keeping track of cell states C_t and $\tilde{C}_t$ shown in Figure 18. The relationships between those variables are governed by the following set of equations: $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$; $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$; $\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$; $C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$; $o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$; and $h_t = o_t * \tanh(C_t)$.

Lastly, bidirectional LSTMs (BiLSTMs) (Schuster and Paliwal, 1997) generalize LSTMs by allowing for information to pass from both forward and backward directions. This is achieved by having two LSTM layers, one corresponding to each direction.

We aimed to improve ResRoFA-MT by allowing for the event representation to be generated by either RNNs, LSTMs or BiLSTMs. The ResRoFSeqRNN-MT model replaces the mean aggregation layer in ResRoFA-MT with RNN blocks, in which the hidden layer output corresponding to the last time step acts as the event embedding. The ResRoFSeqLSTM-MT model is similar to ResRoFSeqRNN-MT, with RNN blocks replaced by LSTM blocks. Figure 19 provides a visual of these architectures. In addition, an architecture which stacks three LSTM layers (ResRoFSeqDeepLSTM-MT) is also considered. A diagram for this architecture can be easily derived from Figure 19.

In addition, two Bi-LSTM based models are considered. The ResRoFSeqBiLSTM-MT model adds the hidden layer outputs corresponding to the last time step of the forward and backward LSTMs in order to form an event embedding. On the other hand, the ResRoFSeqBiLSTMDense-MT model concatenates these hidden layer outputs, and pass it to a dense layer to preserve the output length; the dense layer output is the event embedding. Diagrams for these architectures can be found in Figure 20 and 21.

4.4.2 Applying Attention

The attention mechanism (Vaswani et al., 2017), which can be loosely defined as the assignment of weights to hidden states, has been proved effective in many recent natural language processing (NLP) applications. The weights (or "attention") resemble how humans tend to focus on specific parts of an input, rather than viewing the input as a whole. Specifically, for a sequence of inputs $(x_1, x_2, \ldots, x_T)$, the hidden layers produce a sequence of hidden states $(h_1, h_2, \ldots, h_T)$. Given contextual information for the encoder c_t, the attention score for the hidden state h_t can be computed as $\alpha = \text{softmax}(e_1, e_2, \ldots, e_T)$ where $e_t = f(c_t, h_t)$ (f is usually chosen to be a non-linear function). The attention mechanism is applied to the hidden states by the summation $\sum_{t=1}^{T} \alpha_t h_t$, which becomes the representation that is pushed to the next layer of the model. When there is no given contextual information c_t, h_t could be used in place of c_t, resulting in a mechanism generally known as self-attention.

We experimented with introducing self-attention mechanisms to the architecture, which can be included after a BiLSTM layer or directly after the embedding layer. The

former leads to a model called ResRoFSeqBiLSTMAt-MT, which is a modification of ResRoFSeqBiLSTM-MT in which the event embedding is obtained by applying an additive attention mechanism (Bahdanau, 2015) to the concatenation of the forward and backward hidden layers. The latter leads to a model called ResRoFSeqAt-MT, in which the individual event participant embeddings are immediately followed by an additive attention mechanism. The diagrams for these architectures can be found in Figure 7 and 22. The attention calculation for Figure 22 is the same as that for Figure 7, and hence is omitted for easier view.

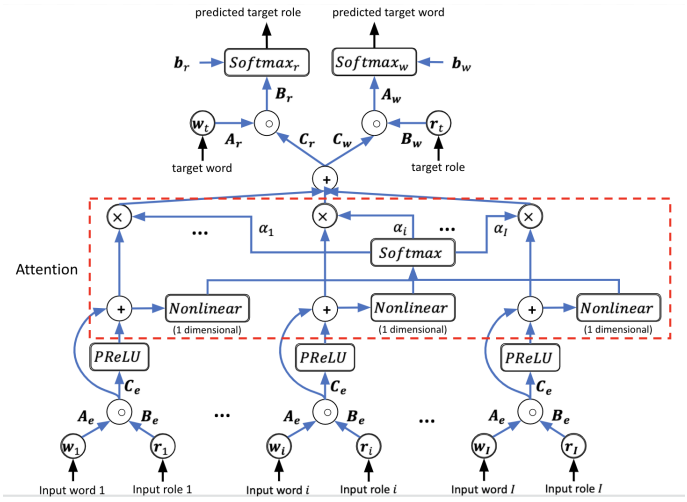

Figure 7: ResRoFSeqAt-MT architecture

4.4.3 Applying CNN

Experimenting with Computer Vision techniques such as Convolution based architectures help provide an alternative perspective of modelling. By stacking Conv1D and MaxPool1D layers, the main idea is to convolve or extract information from the event rather than applying a mean aggregation. Convolution neural networks are known to be attractive for reasons such as weight sharing and exploiting spatial information. For brevity, this model was discontinued due to its poor performance in semantic role labelling and thematic scores, but a diagram for the architecture can be found in Appendix 23.

4.5 Modification of Activation Function

It has been mentioned that one of the differences between Hong et al.'s (2018) ResRoFA-MT model and Tilk et al.'s (2016) NNRF model is the introduction of parametric ReLU (PReLU) layers which provides a parametric weighted mean of the role-filler embeddings. Using the notations in Figure 8, the parameters learned by the PReLU layers in ResRoFA-MT are $a_{i,j}$ for $i \in \mathbb{Z}^+, i \leq 6$ (number of word-role pairs in each input) and $j \in \mathbb{Z}^+, j \leq 256$ (dimension of embeddings).

Considering that the ResRoFA-MT inputs are not ordered, we thought it might be a good idea for the PReLU block corresponding to each time step to learn the same parameters. In particular, we want our learned weights to satisfy $\alpha_{i,j} = \alpha_{k,j}$ for all i, k, j.

This means that instead of learning $6 \times 256 = 1,536$ parameters, we are learning only 256 parameters.

The above-mentioned modification results in an architecture called ResRoFAShared PReLU-MT. Also, we experimented with a special case of ResRoFASharedPReLU-MT with $\alpha_{i,j} = 0.3$ (the default parameter for Leaky ReLU in Keras) for all i, j. This architecture is referred to as ResRoFALeakyReLU-MT. Lastly, as the original implementation of ResRoFA-MT seems to induce ordering due to different parameters learned in each time step, we decided to also train ResRoFA-MT on the sequentially preprocessed data described in Section 3 in an architecture called ResRoFASeq-MT.

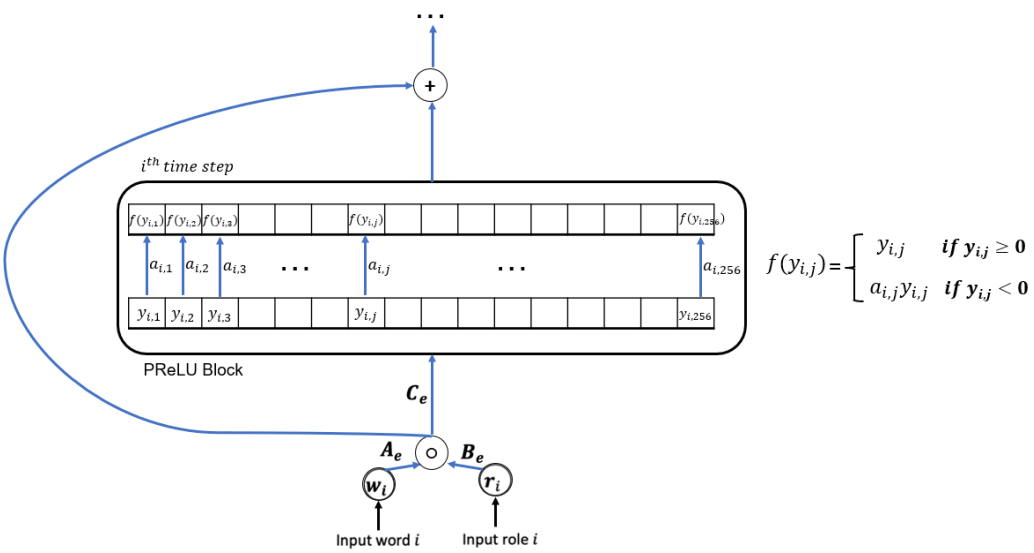

Figure 8: Zooming into a PReLU layer

4.6 Modification of Target Objective

In Hong et al.'s (2018) original implementation of ResRoFA-MT, the overall loss function is defined as $\mathcal{L} = \mathcal{L}^{(w)}(C, r_t) + \alpha\mathcal{L}^{(r)}(C, w_t)$ with $\alpha = 1$. Here, $\mathcal{L}^{(w)}(C, r_t)$ and $\mathcal{L}^{(r)}(C, w_t)$ denote the usual cross entropy loss for the word and role classification task, respectively. Having $\alpha = 1$ can be seen as providing a balance between these two tasks. We decided to experiment with different $\alpha \neq 1$ parameters to see whether this may result in an improvement of ResRoFA-MT. In particular, we tried varying α to be one of $\{0.5, 0.75, 1.5, 2\}$. It can be easily deduced from the loss function form that $\alpha > 1$ causes the model to focus more on role prediction, and $\alpha < 1$ causes the model to focus more on word prediction.

Besides varying the loss function, we can modify the target objective by adding or removing tasks in the multi-task learning setting. For example, we may find new sets of tasks (other than word and role predictions) which correlate better with thematic fit and allow for parameter estimations using a large amount of available data. This is definitely an interesting direction for future research.

4.7 Results and Discussion

In this section, we discuss the the results on the 10% and 1% dataset on both V1 and V2. Note that only high performing models were trained on the 10% data set. The 10% dataset

experiments (available in Figure 9) were trained using Python 3.7 and Tensorflow 2.3.0 in Google Cloud VM environment with 26GB RAM and T4 GPU. For the 1% experiments, these were trained using the Google Colab environment; additional results can be found in Appendix 3, 4 and 5.

Model (10% v1 data)	Optimizer + Learning Rate	Val Role Accuracy	Val Word Accuracy	Pado-All	McRae-All
ResRoFA-MT	Adam 0.01	94.40%	9.20%	45.1	31.9
ResRoFBeg-MT	Adam 0.01	94.40%	9.20%	18.3	16.8
ResRoFSeqAt-MT	Adam 0.001	87.80%	9.50%	52.3	39.5
ResRoFALeakyReLU-MT	Adam 0.01	88.80%	9.50%	34	38.1
ResRoFASharedPReLU-MT	Adam 0.01	88.70%	9.40%	36.3	37.3
Model (10% v2 data)	**Optimizer + Learning Rate**	**Val Role Accuracy**	**Val Word Accuracy**	**Pado-All**	**McRae-All**
ResRoFA-MT	Adam 0.01	97.20%	15.10%	48	41.3
ResRoFBeg-MT	Adam 0.01	97.30%	15.10%	18.8	18.5
ResRoFSeqAt-MT	Adam 0.001	93.50%	15.80%	54.1	38.6
ResRoFALeakyReLU-MT	Adam 0.01	N/A			
ResRoFASharedPReLU-MT	Adam 0.01				

Figure 9: Selected model results on 10% V1 and V2 data

4.7.1 Modified Baseline Architectures

The ResRoFBeg-MT model was trained on the 10% data whilst both ResRoFWD-MT and ResRoFDense-WT were only trained using the 1% data. Initially, the ResRoFBeg-MT model showed potential in the 1% V1 data; with identical role/word accuracy scores against the baseline, along with thematic scores 26.4 and 16.5 for Pado-All and McRae-All. However for the 10% V1 data, the Pado-All and McRae-All scores dropped to 18.3 and 16.8 in Figure 9. As for semantic role labelling performance, it performed identically to the baseline model at 94.4% and 9.2% for role/word accuracy.

As for the ResRoFWD-MT model, it initially performed extremely well for recommendation tasks with 99.9% role accuracy using the 1% V1 dataset, yet it consistently performed poorly in thematic fit scores with negative values. The residual layer had originally improved the thematic scores from -6.7 to 10.7 (Pado-All) and 7.3 to 9.7 (McRae-All) using the 1% V1 data. However, running the model on 10% data showed both Pado-All and McRae-All scores dropping significantly to 3.0.

The ResRoFDense-MT model did not perform well for the 1% V1 data with only 2.1 and 7.1 Pado-All/McRae-All thematic scores. We found that increasing the number of dense layers from one to two marginally increases its scores to 5.7 and 7.3 Pado-All/McRae-All. Interestingly, our results suggests that a single dense layer may not fully capture an event embedding representations; and more complex architectures using dense layers may yield better results.

In all the modified baseline models, residual learning is applied. It's effects are known to reduce the challenges of vanishing gradients when computing factorized tensors (He et al., 2016). For models ResRoFBeg-MT, ResRoFWD-MT and ResRoFDense-WT, we found that adding residual networks may improve thematic fit correlation, but may not affect semantic role labelling.

4.7.2 Attention and PReLU Architectures

The models ResRoFSeqAt-MT, ResRoFALeakyReLU and ResRoFASharedPReLU recorded promising results on thematic fit scores. In particular, ResRoFSeqAt-MT performed strongly on all metrics except role accuracy; with 9.5% word accuracy and 52.3/39.5 for Pado-All/McRae-All scores. Our results suggests that the positional weightings computed for each tasks in ResRoFSeqAt-MT are beneficial, allowing more important inputs to be focused. This prompted us to dive deeper to try different variations of Attention mechanisms in the next section.

Interestingly, the ResRoFSeqAt-MT model recorded lower role accuracy scores in both V1 and V2 datasets. Note that the drop in role accuracy from 94.4% to 87.8% was more prominent in the 10% V1 data; as compared to 97.2% to 93.5% in the 10% V2 data. To investigate why, the baseline model was also trained using sequential data. The sequential baseline model tend to perform better in thematic scores but not role accuracy. This suggests that input word ordering alone can affect semantic role labelling and thematic fit scores.

As for the PReLU models, the thematic scores for ResRoFALeakyReLU and ResRoFASharedPReLU decreased from the baseline scores to 34 and 36.3 (Pado-All), 38.1 and 37.3 (McRae-All) in Figure 9. This is likely due to having less parameters and no positional weightings as described in the original paper (Hong et al., 2018). The decrease in thematic fit scores is possibly due to the learning rate being too high; combined with indirect modelling of tasks between predicting role/word accuracy and thematic fit using the PReLU function. Furthermore, it is observed that forcing the learnable parameter to be the same in PReLU across time steps may increase thematic fit scores at the expense of lower role accuracy.

5 Further Modifications

5.1 Modification of Attention Mechanism

Bahdanau's (2015) additive mechanism is just one of many attention mechanisms used in the literature. A list of all self-attention based models we experimented, which are all based on the literature, can be found in Table 2.

Mechanism	Scoring function	Learnable parameters	Model name
Additive (Bahdanau, 2015)	$e_t = \tanh(Wh_t + b)$	W, b	ResRoFSeqAt-MT
Location-based (Luong, 2015)	$e_t = Wh_t + b$	W, b	ResRoFSeqAtLoc-MT
General (Luong, 2015)	$e_t = h_t^\intercal W h_t$	W	ResRoFSeqAtGen-MT
Dot product (Luong, 2015)	$e_t = h_t^\intercal h_t$	None	ResRoFSeqAtDot-MT
Scaled dot product (Vaswani, 2017)	$e_t = h_t^\intercal h_t / \sqrt{n}$	None	ResRoFSeqAtScaledDot-MT

Table 2: Summary of self-attention based architectures and their scoring functions

In addition, an extension of the attention mechanism has shown that the introduction of task-specific attention increases generalization and performance of the tasks at hand in multi-task model settings (Liu et al., 2019). For a task l, the contextual information c_{tl} together with the hidden state information h_t forms a set of attention weights $\alpha_l = \text{softmax}(e_{1l}, e_{2l}, ..., e_{Tl})$, where $e_{tl} = f(c_{tl}, h_t)$. According to Liu et al. (2019), task-specific attention allows the model to give attention to hidden layers in various ways depending on the task, thus resulting in an increased model generalization and performance. Based on this idea, an architecture called ResRoFSeqTargAt-MT modifies ResRoFSeqAt-MT by having one attention mechanism for each task (i.e. one for word prediction and one for role prediction). Here, for the word prediction task, the contextual information is a target role one-hot encoding vector, and for the role prediction task, the contextual information is a target word one-hot encoding vector. The diagram for this architecture can be found on Figure 10.

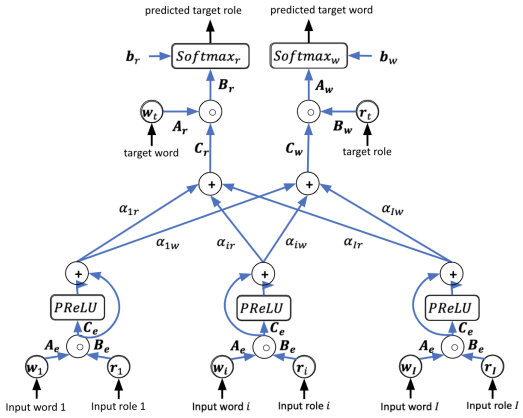

Figure 10: ResRoFSeqTargAt-MT architecture

5.2 Combining Modifications

In this section, we wanted to see the impact of the combination of modification to the activation function combined with target word input timing, and the activation function combined with non-sequential inputs. For all the models created by this combination, `shared axes = [1]` was added to the PReLU layers to convert it to a Shared PReLU model with 256 parameters.

5.3 Results and Discussion

Results of modifying the attention scoring function and the introduction of task-specific attention are shown in Figure 11. In terms of validation role accuracy, the baseline model dominated the other models when all models were trained with 10% of V1 and V2 data. The ResRoFSeqAT-MT model had the highest validation word accuracy among all the other sequential attention model variants and the baseline. In terms of the Pado-All thematic fit evaluation, the ResRoFSeqTargAt-MT consistently performed well for both 10% V1 and V2 data, hinting that task-specific attention allowed the model to capture more of the essential information of the input by utilizing the reference about the specific task. However, regarding the McRae-All thematic fit evaluation, the baseline performed better than the other sequential attention models when models were trained with V2

Model (10% v1 data)	Optimizer + Learning Rate	Val Role Accuracy	Val Word Accuracy	Pado-All	McRae-All
ResRoFA-MT (Baseline)	Adam 0.01	94.40%	9.20%	45.1	31.9
ResRoFSeqAt-MT	Adam 0.001	87.80%	9.50%	52.3	39.5
ResRoFSeqAtDot-MT	Adam 0.001	88.10%	9.30%	50.6	39.5
ResRoFSeqAtScaledDot-MT	Adam 0.001	88%	9.30%	50.5	38.5
ResRoFSeqAtGen-MT	Adam 0.001	87.80%	9.30%	52.4	40.5
ResRoFSeqAtLoc-MT	Adam 0.001	87.50%	9.30%	49.2	38
ResRoFSeqTargAt-MT	Adam 0.001	87.60%	9.40%	53.1	37.5
Model (10% v2 data)	**Optimizer + Learning Rate**	**Val Role Accuracy**	**Val Word Accuracy**	**Pado-All**	**McRae-All**
ResRoFA-MT (Baseline)	Adam 0.01	97.20%	15.10%	48	41.3
ResRoFSeqAt-MT	Adam 0.001	93.50%	15.80%	54.1	38.6
ResRoFSeqAtDot-MT	Adam 0.001	93.60%	15.70%	56	37.9
ResRoFSeqAtScaledDot-MT	Adam 0.001	94%	15.70%	55.9	38.8
ResRoFSeqAtGen-MT	Adam 0.001	93.60%	15.70%	54.3	39.1
ResRoFSeqAtLoc-MT	Adam 0.001	93.10%	15.70%	56.9	33.2
ResRoFSeqTargAt-MT	Adam 0.001	93.40%	15.70%	56.9	38.9

Figure 11: Attention extension results

data. Noting that the superiority of the baseline in terms of V2 McRae-All evaluation is relatively small compared to the differences in Pado-All and the V1 data McRae-All evaluation, and considering the fact that McRae-All is an unbalanced evaluation dataset, the baseline results here may have occurred due to chance and/or the baseline trained on the V2 dataset favoring a certain type of instance. To confirm this, we look deeper into the individual role-level thematic fit performance of the representative models ResRoFA-MT, ResRoFSeqAt-MT, and ResRoFSeqTargAt-MT that were trained on V1 and V2 (Figure 12).

Model (10% v1 data)	mcrae-ARG0	mcrae-ARG1	mcrae-LOC	mcrae-MNR	mcrae-all	pado-ARG0	pado-ARG1	pado-ARG2	pado-All
ResRoFA-MT	0.3006	0.3353	0.4984	0.3999	0.3191	0.3838	0.5382	0.5786	0.4508
ResRoFSeqAt-MT	0.3827	0.4176	0.4454	0.3768	0.3952	0.4979	0.5763	0.6066	0.523
ResRoFSeqTargAt-MT	0.3638	0.3919	0.446	0.3928	0.3746	0.5032	0.5817	0.6173	0.531
Model (10% v2 data)	**mcrae-ARG0**	**mcrae-ARG1**	**mcrae-LOC**	**mcrae-MNR**	**mcrae-all**	**pado-ARG0**	**pado-ARG1**	**pado-ARG2**	**pado-All**
ResRoFA-MT	0.4562	0.3576	0.3909	0.3551	0.4133	0.4399	0.5515	0.6743	0.4795
ResRoFSeqAt-MT	0.3682	0.4232	0.3761	0.3018	0.3856	0.5557	0.5724	0.3479	0.5412
ResRoFSeqTargAt-MT	0.3779	0.4208	0.3543	0.3207	0.3888	0.5859	0.6123	0.3161	0.5687

Figure 12: Role-specific thematic fit scores

We notice that in most cases for these models, the shift from V1 to V2 increased thematic fit performance for examples that included ARG0 [agent-like] and ARG1 [patient-like], but on the other hand thematic fit performance decreased for examples that included other roles such as ARGM-LOC [location], ARGM-MNR [manner] and ARG2 [instrument], except for the combination of ResRoFA-MT and ARG2. In general, the performance declination could be caused due to a known issue in He-SRL, which is that it confuses ARG2 with other roles such as ARGM-LOC and ARGM-MNR. For the ResRoFA-MT though, there was an increase in performance, which could be thought that since ResRoFA-MT had fewer parameters and a simpler non-sequential architecture, it was less likely to overfit to invalid data, compared to the ResRoFSeqAt-MT and ResRoFSeqTargAt-MT models, which consists of additional parameters from the attention mechanism and sequential characteristic.

But in general the increase in thematic fit performance for examples with ARG0 and ARG1 was greater in magnitude compared to the other roles, which led to an improvement in thematic fit most of the time.

Results of modifying the activation function to SharedPReLU are shown in Figure 13. For the sequential attention models, thematic fit performance was better when the original PReLU was used for the activation function, but for the ResRoFSeqTargAt-MT, the McRae-All was slightly better with the SharedPReLU activation function. Again, this could be thought of as a phenomenon that happened by chance, or due to the McRae-All being an unbalanced patient/agent dataset. More analysis and experiments would need to be conducted to confirm this.

Model (10% v1 data)	Optimizer + Learning Rate	Val Role Accuracy	Val Word Accuracy	Pado-All	McRae-All
ResRoFA-MT (Baseline)	Adam 0.01	94.40%	9.20%	45.1	31.9
ResRoFA-SharedPReLU-MT	Adam 0.01	88.70%	9.40%	36.3	37.3
ResRoFBeg-MT	Adam 0.01	94.40%	9.20%	18.3	16.8
ResRoFBegSharedPReLU-MT	Adam 0.01	94.50%	9.10%	4.2	12.7
ResRoFSeqAt-MT	Adam 0.001	87.80%	9.50%	52.3	39.5
ResRoFSeqAtSharedPReLU-MT	Adam 0.001	87.40%	9.00%	49.8	38.6
ResRoFSeqAtGen-MT	Adam 0.001	87.80%	9.30%	52.4	40.5
ResRoFSeqAtGenSharedPReLU-MT	Adam 0.001	87.40%	9.00%	51.5	39.4
ResRoFSeqTargAt-MT	Adam 0.001	87.60%	9.40%	53.1	37.5
ResRoFSeqTargAtSharedPReLU-MT	Adam 0.001	87.00%	9.00%	46.1	37.9

Figure 13: Activation extension results

6 Ethical Considerations

Even if a model produces promising results in terms of thematic fit, ethical considerations have to be made for both the data and the models to prevent consequences that could possibly harm others. These consequences include bias, privacy issues, security threats, noncompliance to laws and regulations, transparency issues, and unintended use cases. Below we describe some consequences that may arise in our research setting in terms of the data and the model, and some possible actions we could take for prevention.

6.1 Bias

Regarding data, bias could be introduced from historically biased data, specifically regarding sensitive attributes such as race and gender. This is usually introduced from the uneven attribute proportions in the data, and it could affect how the model makes predictions. For example in our setting, it is easy to imagine a case where for the predicate "work", "man" could be given higher probability to be the subject than "woman" due to historical reasons. In addition, only the most common 50,000 words were extracted from the data, which biases the distribution of words, because they do not become representative of the word distribution in real life. The limitation in the number of words also causes the thematic fit evaluation set to have less samples, since samples that do not include the words are filtered out. Therefore, the data that the model reads in is biased, and the evaluation of the model has some bias as well.

Regarding the model, if the model architecture is designed based on a model designer's strong assumptions, it could introduce bias in the predictions. This includes the physical connections between layers, activation functions, loss functions, and any other hyperpara meter-related setting that the model designer decides on.

To mitigate bias in data, a simple extension of allowing more words to be included into the data could be helpful, as well as filtering out sensitive words that could introduce discrimination. The model should be constructed so that the model designer's assumptions do not dominate the model, and prevent the model from being trained flexibly.

6.2 Privacy Concerns

As mentioned earlier, we use the RW-eng data to train our models. The RW-eng data uses the ukWaC and BNC data sets as data sources, and if there are privacy concerns embedded in these original data sources, the RW-eng data could also be vulnerable to them. Unfortunately, ukWaC contains personal pages, blogs, and postings, while BNC contains school and university essays in their data, which could introduce privacy concerns for the RW-eng data as well. Since the RW-eng, ukWaC, and BNC datasets are all available online, and in the case they include information that is private, it is challenging to remove that information from the web. Even so, measures such as removing pronouns could be taken to ensure privacy for the dataset and model in our research setting.

6.3 Security Threats

This falls down to where and how the data and model files are stored. Currently, data is stored within Google Cloud services, or on local machines. Since Google Cloud services consider security as a core competency, although it is a black box, it should be much more secure than storing data locally. Storing data locally allows the team to conduct experiments and scripts on personal machines, making the project proceed more efficiently, but introduces vulnerabilities specifically regarding the manipulation of data by malicious parties or viruses. The same issue arises for the weights of the saved models; weights saved locally could be manipulated causing models to produce incorrect predictions. Losing efficiency in a short-term project is punishing, but efforts should be made to keep the majority of the project running on secure locations such as Google Cloud.

6.4 Laws and Regulations

Currently there are no laws and regulations that the project is not compliant with, to the knowledge of the team. In the case there is actual confidential information that harms individuals included in the dataset, swift actions should be taken to remove all of the applicable data and models, and take action to support all that were affected.

6.5 Transparency

The mechanism behind the creation of the RW-eng data could be obscure, since the RW-eng data is created using the pre-existing semantic role labeler SENNA, where the role

labeling algorithm could be difficult to decipher. In addition, there is a lack of transparency in the evaluation of thematic fit, due to an indirect link between the task the model is trained for and thematic fit. The usage of neural networks for our model also hinders our understanding of the logic behind the predictions. Understanding SENNA and clearly describing the steps of the role labeling will improve transparency, and cumulatively making changes to the model architectures will help with the understanding of how models are functioning, leading to better transparency.

6.6 Unintended Use Cases

The RW-eng data could possibly be used for fake news generator training, and for obtaining personal information in the case personal information was included in the previously mentioned ukWaC and BNC data sets. It is extremely difficult to prevent fake news generator training, but the removal of personal information is an action that could be taken.

7 Conclusion and Future Work

7.1 Conclusion

In this report, we explore different modifications to ResRoFA-MT, a state-of-the-art model for thematic fit developed by Hong et al. (2018). These modifications are made on the (1) input style; (2) target word input timing; (3) aggregation function; (4) non-sequential input; (5) activation function; and (6) target objective. We observe that the first three modifications consistently result in lower thematic fit scores as compared to the baseline. The fourth modification, which involves modifying the model input to be sequential and employing the attention mechanism, sees a significant increase in thematic fit scores for both V1 and V2 data, except the McRae score on V2 data. However, this model achieves a lower role accuracy than the baseline. Further exploration on the attention mechanism results in a multi-task attention model called ResRofSeqTargAt-MT, which achieves the highest Pado score for both V1 and V2 data. The fifth modification generally results in lower thematic fit scores, while the sixth modification does not significantly impact the model performance.

7.2 Future Work

Additional experiments and analysis needs to be conducted to provide stronger reasoning for the results of our model, such as investigating more deeply into why in certain models McRae-All evaluation performs worse than the baseline, examining why sequential models generally have lower role accuracies, and training all models on the V1 and V2 10% data to provide more consistent and complete results. Furthermore, as new directions into this research, exploring new target objectives and adding or removing tasks, and collaborating with Team 1 to combine word embeddings and architectures could improve thematic fit performance.

8 Acknowledgements

Our contributions for this project are as follows:

- Timothy Huang: Environment Setup, Code Conversion, Model Training
- Jay Ng: Code Conversion, Model Implementation, Architecture Designing, Literature Review, Model Training
- Yuki Nishimura: Model Implementation, Sequential Preprocessing, Architecture Design, Ethical Considerations, Team 1 Collab, Literature Review, Model Training
- Jonathan Santoso: Code Conversion, Evaluation Metrics Modification, Model Training
- Kevin Wibisono: Literature Review, Model Implementation, Architecture Designing, Slide Making, Model Training

In addition, we would like to express our deepest gratitude to Professor Yuval Marton from Bloomberg CTO Office, Professor Asad Basheer Sayeed from the University of Gothenburg, as well as Professor Smaranda Muresan from Columbia University for supervising and guiding us throughout the entire project.

9 Bibliography

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Representations*.

Marco Baroni and Alessandro Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics* 36(4):673–721.

Klinton Bicknell, Jeffrey L Elman, Mary Hare, Ken McRae, and Marta Kutas. 2010. Effects of event knowledge in processing verbal arguments. *Journal of memory and language* 63(4):489–505.

Xavier Carreras and Lluis Marquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*. Association for Computational Linguistics, Ann Arbor, Michigan, pages 152–164.

Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, and others. 2016. Wide and deep learning for recommender systems. In *Proceedings of the Recsys*. 7–10.

Todd R Ferretti, Ken McRae, and Andrea Hatherell. 2001. Integrating verbs, situation schemas, and thematic role concepts. *Journal of Memory and Language* 44(4):516–547

Clayton Greenberg, Vera Demberg, and Asad Sayeed. 2015. Verb polysemy and frequency effects in thematic fit modeling. In *Proceedings of the 6th Workshop on Cognitive Modeling and Computational Linguistics*. Association for Computational Linguistics, Denver, Colorado, pages 48–57.

Edward Grefenstette and Mehrnoosh Sadrzadeh. 2015. Concrete models and empirical evaluations for the categorical compositional distributional model of meaning. *Computational Linguistics* 41(1):71–118.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Association for Computational Linguistics, Las Vegas, Nevada, pages 770–778.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep Semantic Role Labeling: What Works and What's Next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, pages 473-483.

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018. Jointly predicting predicates and arguments in neural semantic role labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Melbourne, Australia, pages 364–369.

Sepp Hochreiter and Jurgen Schmidhuber. 1997. Long short-term memory. In *Neural Computation*, vol. 9, no. 8, pages 1735–1780.

Xudong Hong, Asad Sayeed, and Vera Demberg. 2018. Learning distributed event representations with a multi-task approach. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*. Association for Computational Linguistics, New Orleans, Louisiana, pages 11-21.

Shikun Liu, Edward Johns, and Andrew J. Davison. 2019. End-to-end multi-task learning with attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, California, pages 1871–1880.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceddings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 1412–1421.

Ken McRae, Mary Hare, Jeffrey L Elman, and Todd Ferretti. 2005. A basis for generating expectancies for verbs from nouns. In *Memory & Cognition*, vol. 33, no. 7, pages 1174–1184.

Christopher Olah. 2015. Understanding LSTM networks. Obtained from https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Ulrike Pado. 2007. *The integration of syntax and semantic plausibility in a wide-coverage model of human sentence processing*. Ph.D. thesis, Saarland University.

Asad Sayeed and Vera Demberg. 2014. Combining unsupervised syntactic and semantic models of thematic fit. In *Proceedings of the first Italian Conference on Computational Linguistics (CLiC-it 2014)*.

Asad Sayeed, Pavel Shkadzko, and Vera Demberg. 2018. Rollenwechsel-English: a large-scale semantic role corpus. In *Proceedings of the Eleventh edition of the Language Resources and Evaluation Conference*. European Language Resources Association, Miyazaki, Japan, pages 3087–3091.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. In *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pages 2673–2681.

Ottokar Tilk, Vera Demberg, Asad Sayeed, Dietrich Klakow, and Stefan Thater. 2016. Event participant modelling with neural networks. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Austin, Texas, pages 171–182.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceddings of the International Conference on Neural Information Processing Systems*. Curran Associates, Long Beach, California, pages 6000–6010.

A Appendix

A.1 Additional Figures

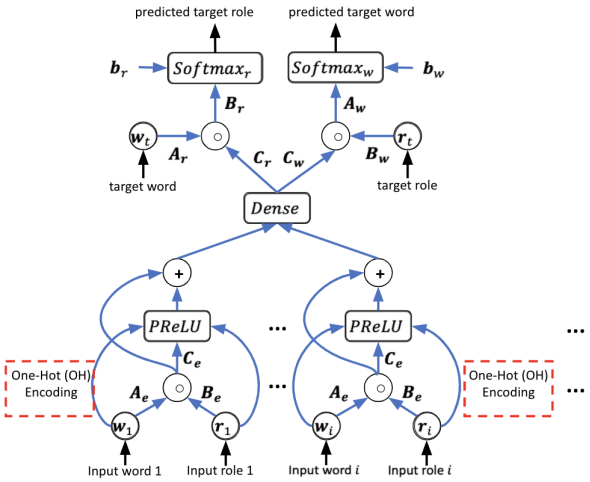

Figure 14: ResRoFWD-MT architecture

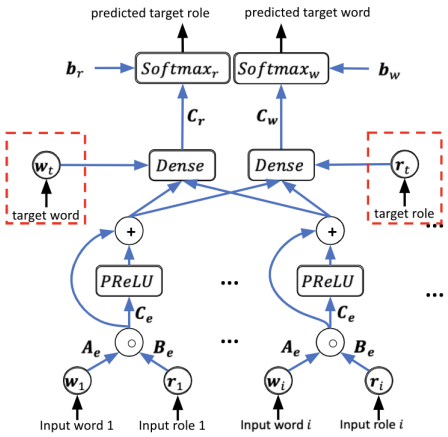

Figure 15: ResRoFBeg-MT Model

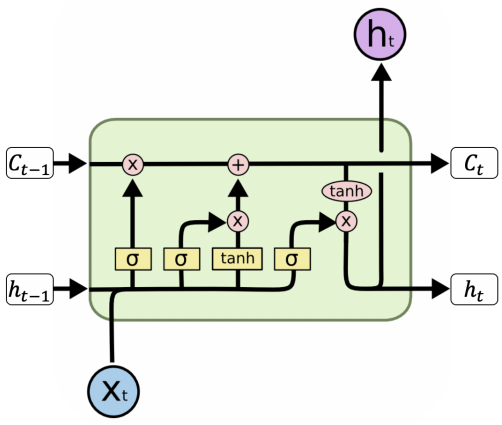

Figure 18: Zooming into an LSTM cell (Olah, 2015)

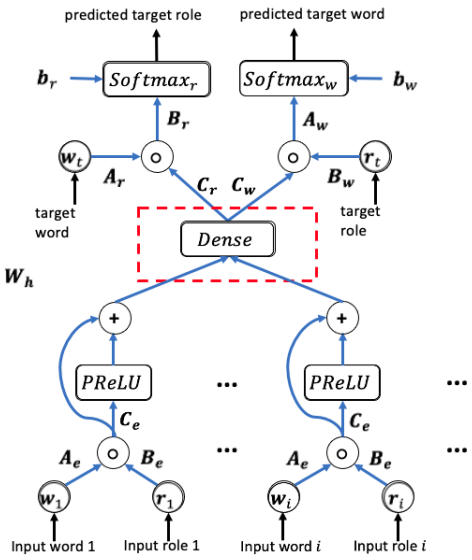

Figure 16: ResRoFDense-MT architecture

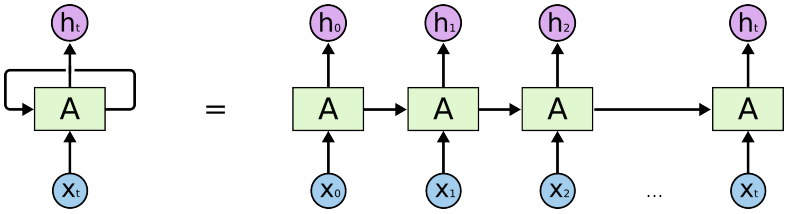

Figure 17: An unrolled RNN layer (Olah, 2015)

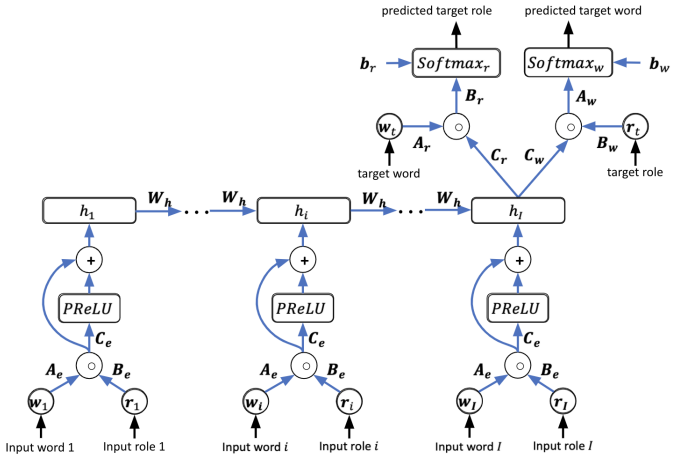

Figure 19: ResRoFSeqRNN-MT (with $h_i = RNN_i$) and ResRoFSeqLSTM-MT (with $h_i = LSTM_i$) architectures

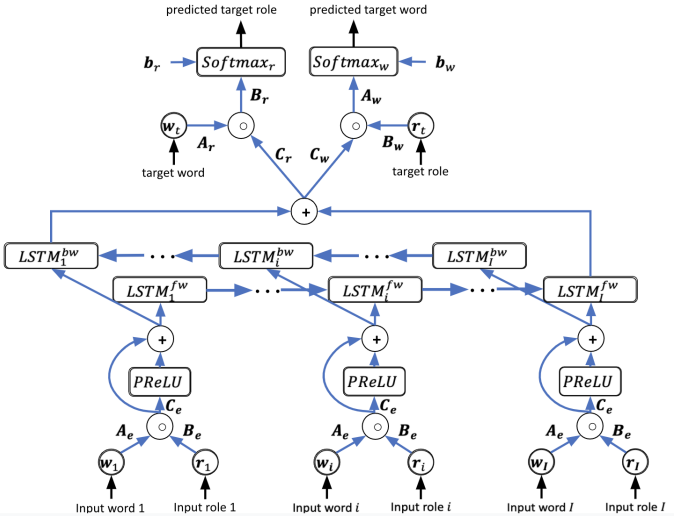

Figure 20: ResRoFSeqBiLSTM-MT architecture

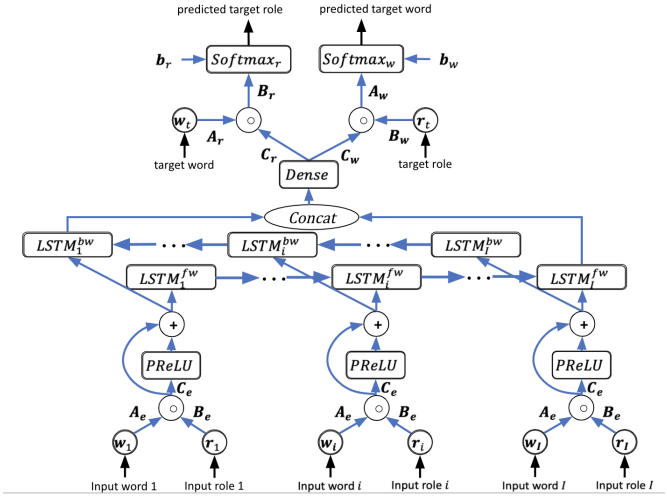

Figure 21: ResRoFSeqBiLSTMDense-MT architecture

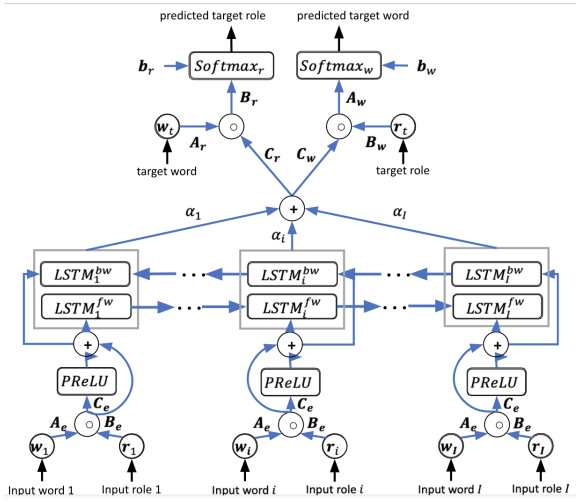

Figure 22: ResRoFSeqBiLSTMAt-MT architecture

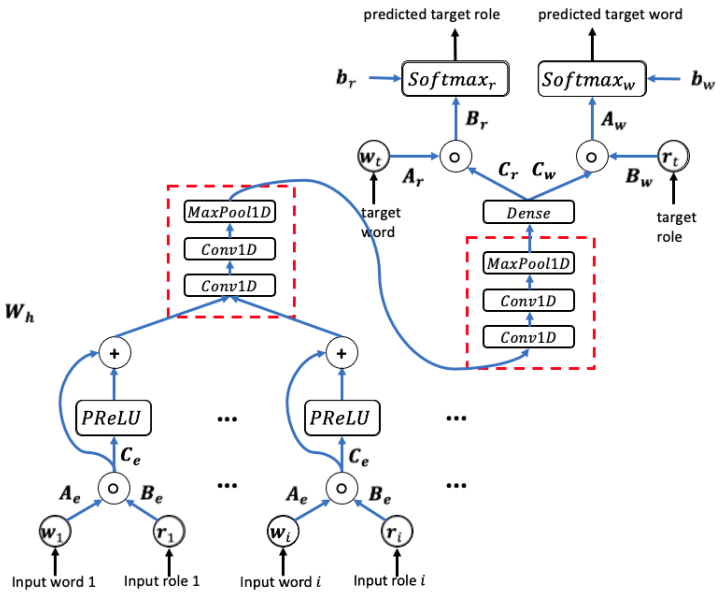

Figure 23: ResRoFSeqConv-MT architecture

A.2 Additional Result Tables

Model (1% v1 data)	Optimizer + Learning Rate	Val Role Accuracy	Val Word Accuracy	Pado-All	McRae-All	Max Pado	Max McRae
ResRoFA-MT	Adam 0.01	93.50%	7.40%	27.6	18.8	40.6	24.9
ResRoFDense-MT	Adam 0.01	93.40%	7.20%	2.1	7.1	11.7	9.5
ResRoFBeg-MT	Adam 0.01	93.90%	7.60%	17.9	17.4	40.9	22.3
ResRoFWD-MT	Adam 0.01	99.90%	8.90%	10.7	9.7	13.5	14.8
ResRoFSeqLSTM-MT	Adam 0.01	72.20%	4.10%	18.7	26.8	31.5	30
ResRoFSeqBiLSTMDense-MT	Adam 0.01	84.20%	5.00%	26.9	26.4	29.2	26.4
ResRoFSeqAt-MT	Adam 0.01	86.90%	7.50%	41.3	30.9	47.9	33
ResRoFSeqAt-MT	Adam 0.001	86.20%	7.50%	43.1	27.1	45.2	31.6
ResRoFSeqAtDot-MT	Adam 0.01	89.10%	7.50%	38.7	30.4	49.5	33.7
ResRoFSeqAtDot-MT	Adam 0.001	85.10%	7.30%	41.3	29.3	44.7	30
ResRoFSeqAtScaledDot-MT	Adam 0.01	86.50%	7.80%	49.3	29	50.5	32.3
ResRoFSeqAtScaledDot-MT	Adam 0.001	86.00%	7.40%	40	24.9	43.3	28.3
ResRoFSeqAtGen-MT	Adam 0.001	78.40%	7.50%	45.3	31.3	46.9	32.1
ResRoFSeqAtLoc-MT	Adam 0.001	85.90%	7.40%	41	30.1	46.2	30.3
ResRoFSeqTargAt-MT	Adam 0.01	86.80%	7.50%	42.9	31.5	50.8	34.2
ResRoFSeqTargAt-MT	Adam 0.001	85.50%	7.00%	40.7	24.8	40.7	26.4
ResRoFSeqConv-MT	Adam 0.01	65.30%	3.60%	27.4	24.4	34.5	25.8

Table 3: Summary of select model results on 1% V1 data

Model (1% v2 data)	Optimizer + Learning Rate	Val Role Accuracy	Val Word Accuracy	Pado-All	McRae-All	Max Pado	Max McRae
ResRoFA-MT	Adam 0.01	96.40%	13.20%	34	21.1	39.3	25.7
ResRoFDense-MT	Adam 0.01	96.00%	10.90%	9.5	3.7	19.6	6.2
ResRoFBeg-MT	Adam 0.01	96.70%	13.30%	34.2	18.2	34.7	23.2
ResRoFWD-MT	Adam 0.01	99.90%	13.10%	10.2	22.2	14.3	23.8
ResRoFSeqLSTM-MT	Adam 0.01	82.30%	8.40%	30.8	21.5	33.9	29.1
ResRoFSeqBiLSTMDense-MT	Adam 0.01	87.80%	9.10%	29.8	24.5	33.5	27.2
ResRoFSeqAt-MT	Adam 0.01	92.30%	13.20%	50.2	27.9	54.2	31.4
ResRoFSeqAt-MT	Adam 0.001	92.00%	13.20%	42.3	29.3	47.6	27.8
ResRoFSeqAtDot-MT	Adam 0.01	92.20%	13.70%	44.7	30.9	50.8	32.3
ResRoFSeqAtDot-MT	Adam 0.001	82.90%	12.30%	42.9	30.7	50.8	30.8
ResRoFSeqAtScaledDot-MT	Adam 0.01	92.40%	13.40%	50.4	30.6	52.4	32.9
ResRoFSeqAtScaledDot-MT	Adam 0.001	92.10%	13.00%	42.9	28.9	47.6	29.3
ResRoFSeqAtGen-MT	Adam 0.001	82.70%	12.40%	51.3	30	51.3	32.2
ResRoFSeqAtLoc-MT	Adam 0.001	92.10%	13.60%	47.1	29.3	50.2	31.6
ResRoFSeqTargAt-MT	Adam 0.01	92.40%	13.10%	50.7	29.4	55.2	32.1
ResRoFSeqTargAt-MT	Adam 0.001	92.10%	12.50%	50.2	28.3	55.8	31.6
ResRoFSeqConv-MT	Adam 0.01	86.00%	7.60%	19.7	14.5	29.1	26.9

Table 4: Summary of select model results on 1% V2 data

Model (1% v1 data)	Alpha (Loss of role)	Optimizer + Learning Rate	Val Role Accuracy	Val Word Accuracy	Pado-All	McRae-All	Max Pado	Max McRae
ResRoFA-MT	0.5	Adam 0.01	93.30%	7.50%	39.2	27	39.2	27.1
ResRoFA-MT	0.75	Adam 0.01	93.50%	7.30%	24.2	13.3	32	19.3
ResRoFA-MT	1	Adam 0.01	93.50%	7.40%	27.6	18.8	40.6	24.9
ResRoFA-MT	1.5	Adam 0.01	93.50%	7.50%	22.2	15.3	26.2	18
ResRoFA-MT	2	Adam 0.01	93.50%	7.50%	21.7	19.7	29.9	22.2

Table 5: Summary of ResRoFA-MT results with varying α on 1% V1 data