Chapter 3

Methodology

Given a {headline} and {body text} pair, our goal is to determine whether a news article contains an incongruent headline. So we calculate the output probability as an incongruence score in our dissertation.

3.1 Baseline approaches

We present some baseline approaches that have been used to solve the headline incongruence problem. Feature-based ensemble algorithms are frequently utilized due to their simplicity and effectiveness. Among various methods, the XGBoost algorithm has demonstrated superior performance in a variety of prediction tasks (Chen and Guestrin 2016).

To begin, we implemented the XGBoost (XGB) classifier using the features given in the winning model for the FNC-1 competition, such as cosine similarity between the {headline} and {body text}.

3.1.1 XGBoost

XGBoost, which implements gradient boosted decision trees, is a well-recognized and fast algorithm for classification tasks (Chen and Guestrin 2016). We adopted XGBoost as a representative baseline because it was used in the winning model for the stance detection challenge in news headlines.
Extreme gradient boosting, often known as XGBoost, is a well-known gradient boosting technique(ensemble) that improves the performance and speed of tree-based (sequential decision trees) deep learning algorithms. Using an incongruity label instead, we implemented the winning model from this challenge by extracting a feature set consisting of TF-IDF vectors based on word occurrences. Singular values decomposed from these

vectors indicate word-vector similarities between a {headline} and its corresponding {body text}. We call this model XGB.

3.1.2 Recurrent Dual Encoder (RDE)

To compute the similarity between two text inputs, a recurrent dual encoder composed of dual RNNs was used (Lowe et al. 2015). When an RNN encodes a word sequence, each word is passed through a word-embedding layer, which converts a word index to a 300-dimensional vector. Using the final hidden state of each headline and body text RNN, the probability of having an incongruent headline is calculated after the encoding step. In the training objective, the incongruence score is as follows:

$$p(\text{label}) = \sigma((h^{H}_{th})^{T} M h^{B}_{tb} + b),$$

$$L = -\log \boldsymbol{\Pi}^{N}_{n=1} p(\text{label}_n | h^{H}_{n,th}, h^{B}_{n,tb}) \quad \ldots.(3.1)$$

where h^{H}_{th} and h^{B}_{tb} are the last hidden state of each {headline} and {body text} RNN with the dimensionality $h \in R^{d}$. The $M \in R^{d*d}$ and bias b are learned model parameters. σ is the sigmoid function and N is the total number of samples utilized in training.

3.1.3 Convolution Dual Encoder (CDE)

CDE is a CNN-based approach that employs a small set of words as context. Local features are calculated with various-sized filters, and the model determines the maximum value over time. CDE extracts one of the most useful characteristics in this method to see if the article has incongruent headlines. We apply Convolutional Dual Encoder to the headline incongruence problem, following the CNN architecture for text understanding (Kim 2014). We obtained a vector representation $v = \{v_i | i = 1, \cdots, k\}$ for each section of the article using max-over-time pooling after computing convolution with k filters using the headline and body text word sequences as input to the convolutional layer as follows:

$$v_i = g(f_i(W)) \quad \ldots.(3.2)$$

where g denotes the maximum-over-time pooling function, f_i denotes the CNN function with the i-th convolutional filter, and $W \in R^{t*d}$ is a word sequence matrix. Using dual CNNs, we encode the {headline} and {body} text as vector representations.

3.2 Proposed methods

Previous research has shown that long word sequences in news articles reduce performance. Because of RNN's natural property of forgetting knowledge from long-range data, it would be less successful as the length of the word sequences in the text increased. The recurrent neural network used in RDE, for example, is bad at recalling information from the distant past. While CDE learns local word dependencies, the model's normal convolutional filter length prevents it from catching any relationship between words in different places. Because news stories might be very long, the inability to handle long sequences is a key limitation of using typical deep techniques to the headline incongruence problem. Our news corpus has an average word count of 518.97.

As a result, we propose neural architectures that effectively learn hierarchical structures of large text sequences to fill this gap. We also provide a data augmentation strategy that effectively reduces the target content's length while expanding the training set's size. Now we'll discuss the model we've proposed. The AHRDE architecture was created to solve this issue.

3.2.1 Attentive Hierarchical Recurrent Dual Encoder (AHRDE)

Using a two-level hierarchy of RNN architecture, this model divides the text into a list of paragraphs and encodes the complete text input from the word level to the paragraph level. The model learns the relevance of each paragraph in the body text based on the article's headline by including an attention mechanism in the paragraph-level RNN.

In paragraph-level RNNs, we also use bi-directional RNNs to utilize information from the past and future. This model makes use of paragraph structure to address the news article's arbitrary length.

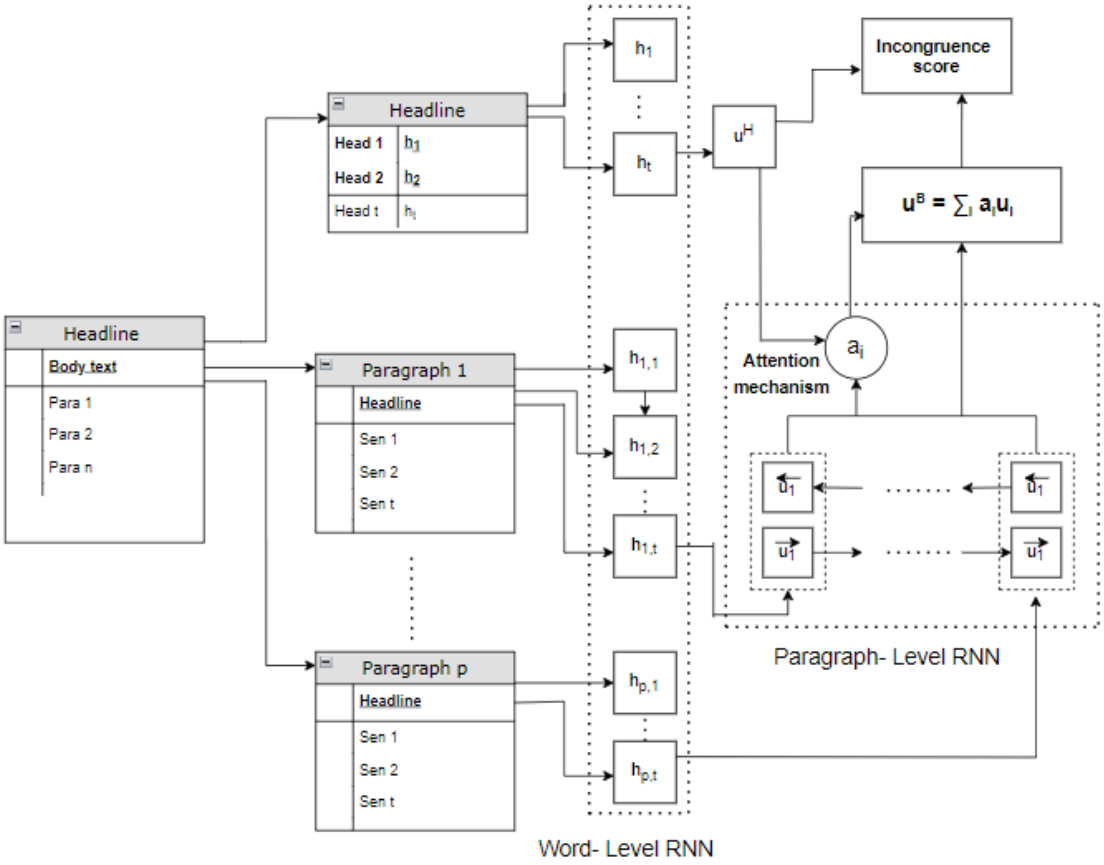

Figure 3.1: Diagram of AHRDE model.

A diagram of the AHRDE model is shown in Figure 3.1. A two-level hierarchy is used to encode the entire text input from the word level to the paragraph level. An attention mechanism can teach the model the importance of each paragraph in the body text based on the headline of the article. The word-level RNN encodes the word sequences $w_p = \{w_{p,1:t}\}$ to $h_p = \{h_{p,1:t}\}$ for each paragraph. The word-level RNN's hidden states are then sent into the next-level RNN, which models a sequence of paragraphs while maintaining order. The hierarchical design, in comparison to RDE,

can learn textual patterns of news items with fewer sequential steps than RNNs. Hierarchical RNNs have the following hidden states:

$$h_{p,t} = f_{\Theta}(h_{p,t-1}, w_{p,t}),$$
$$u_p = g_{\Theta}(u_{p-1}, h_p) \quad \ldots.(3.3)$$

where u_p is the hidden state of the paragraph-level RNN at the p-th paragraph sequence, and h_p is the last hidden state of the word-level RNN of each paragraph $h_p \in \{h_{1:p,t}\}$. Then, because each paragraph of body material corresponds to the title, and each paragraph u_p of {body text} is classified and aggregated according to its correspondence with the {headline}, as follows:

$$s_p = v^T \tanh(W^B_u \; u^B_p + W^H_u \; u^H),$$
$$a_i = \exp(s_i)/\sum_p \exp(s_p),$$
$$u^B = \sum_i a_i u^B_i \quad \ldots.(3.4)$$

where u^B_p denotes the paragraph level RNN's p-th hidden state for learning the representation of body text. The u^H denotes the paragraph-level RNN with the {headline} latest hidden state. The incongruence score is calculated as shown below, using the same training objective as the RDE model:

$$p(\text{label}) = \sigma((u^H)^T M \, u^B + b) \quad \ldots.(3.5)$$

3.2.2 Embedding Recurrent Encoder (ERE)

The AHRDE model encodes text from the word level to the paragraph level using two hierarchical RNNs. When compared to non-hierarchical alternatives like RDE and CDE, the model demands more computation resources for training and inferencing.

The main {body text} of the news is divided into paragraphs, each of which is embedded by averaging the word-embedding vector from its containing words. In other words, ERE calculates h_p in equation (3.3) by averaging the word embedding among paragraph p's words.

$$h_p = \sum_i \text{embedding}(w_i), \quad w_i \subset \text{p-th paragraph.}$$

The embedding procedure is constructed as follows for a given input sequence $x = \{x_{1:t}\}$ with these K vectors:

$$p_k = \text{softmax}((x)^T m_k),$$

$$x_K = \sum^K_{k=1} p_k m_k,$$

$$e = \text{concat}\{x, x_k\} \quad \ldots.(3.6)$$

Dot-product is used to calculate the similarity of the x and each latent topic vector. The softmax function $\text{softmax}(z_k) = e^{zk}/\sum_i e^{zi}$ is then used to normalize the resulting K values, yielding the similarity probability p_k. x_k is recovered by summing over m_k weighted by the p_k after the latent topic probability p_k has been calculated. The final encoding vector e is generated by concatenating this result with the initial encoding vector. Then, using the paragraph-encoded sequence input, a paragraph-level RNN is used to retrieve the final encoding vector for the entire body text.

3.3 Data Augmentation

In data analysis, data augmentation refers to methods for increasing the amount of data available by adding slightly modified copies of current data or creating new synthetic data from existing data. When training a deep learning model, it acts as a regularizer and reduces overfitting. While AHRDE outperforms ERE when there are a few paragraphs, the performance gap narrowed as the paragraph size increased, and ERE received the highest score for exceptionally long input (i.e., a news article containing 19-20 paragraphs). This pattern may be explained by the fact that ERE has a smaller number of trainable parameters than AHRDE.

After using the IT data augmentation strategy, the newly proposed ERE and AHRDE models consistently demonstrated competitive performance regardless of the paragraph size in the {body text}. When the IT approach was used, prediction performance also improved significantly. RDE and CDE benefited the most from the IT technique, achieving results that were comparable to the hierarchical model.

3.3.1 Independent Training (IT) Method

We implement a data augmentation strategy that divides paragraphs in the {body text} and learns the relationship between each paragraph and headline separately called an Independent Training (IT) Method. To determine the incongruence score, each paragraph of a news report is compared to the headline. The maximum value is used to determine the final incongruence score.
The IT technique, as shown in Figure 3.2, calculates the incongruence score for each paragraph based on its link with the news headline.

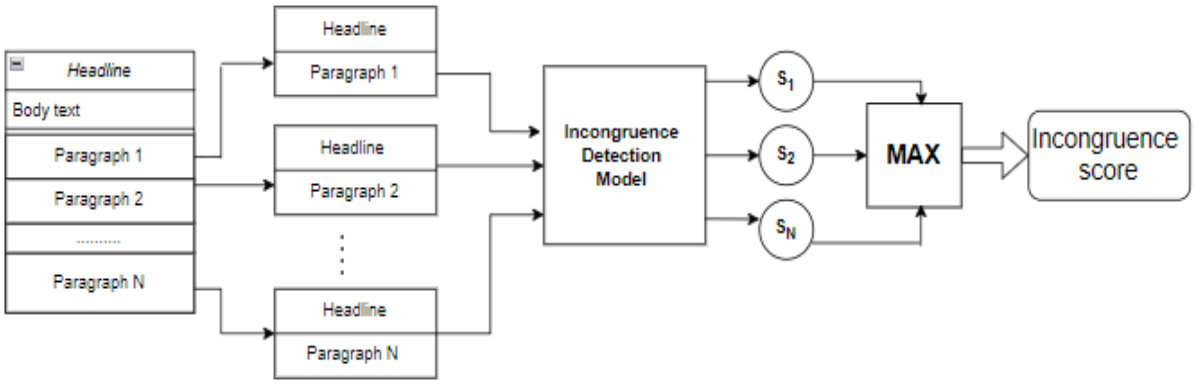

Figure 3.2: Diagram of the independent training method.

To determine the incongruence score, each paragraph of a news report is compared to the headline. The maximum value is used to determine the final incongruence score. The maximum score of incongruence scores is used to produce the final incongruence score for the pair of {headline} and {body text} as follows :

$$p(\text{label}) = \max(s_{1:p}), \quad \ldots.(3.7)$$

where s_p is the incongruence score derived from the p-th paragraph of the {body text} and the {headline}. The maximum score option can better identify news stories that have a paragraph that is completely unrelated to the news headline.
We utilize the paragraph dataset to train models with the IT approach, and incongruence scores are obtained as follows:

• **XGB with IT:** The cosine similarities between the {headline} and the {body text} are calculated using the XGBoost (XGB) classifier. Word-vector similarities between a headline and its associated body text are indicated by singular values decomposed from these vectors. XGB measures the incongruence score for each paragraph to extract features from the headline and each paragraph of the body text.

• **RDE/CDE with IT:** RDE calculates the similarity of two text inputs and turns the result into a 300-dimensional vector. Whereas, CDE extracts one of the most useful aspects to see if the article has incongruent headlines. To explore the relationship between {headline} and {body text}, CDE relies on the number of incongruent words rather than the patterns of implanted paragraphs, such as the order or the insertion place. Both models compare the encoded {headline} to word sequences in each paragraph of {body text}.

• **AHRDE with IT:** These RNN-based techniques outperformed the others in all kinds, which can be related to their capacity to exploit sequential data. To encode the word sequence in each {headline} and {body text}, we used two single-layer GRUs units for the word-level RNN part. For the paragraph-level RNN part, we employed two single-layer bidirectional GRUs units to encode the final hidden state sequences from each word-level RNN of the {headline} and {body text}.
The first-level RNN encodes word sequences for each sentence to encode each paragraph in body text, and the second-level RNN takes a sequence of sentences as input from the first-level RNN and models a sequence of paragraphs while maintaining the order.

• **ERE with IT:** When the amount of news content implanted increases, HRE performance decreases. This is because the number of widely used words increases in response to the length of the sentence. ERE is an RNN-based model, like AHRDE/RDE, which means it uses sequential data as well. ERE initially calculates the mean of word vectors for each sentence to get the incongruence score for each paragraph. RNN then takes the mean word vectors as input and encodes a sequence of sentences.

When the amount of implanted content is little, the IT method and trend become much more obvious. This is because the model can study each portion of the news item more thoroughly when it is separated into many paragraphs. The model's hierarchical architecture allows it to achieve similar effects over the entire text. Multiple systematic encoding can aid the model in remembering the lengthy or complex things that a simple model could overlook.

3.4 Flowchart of the Model

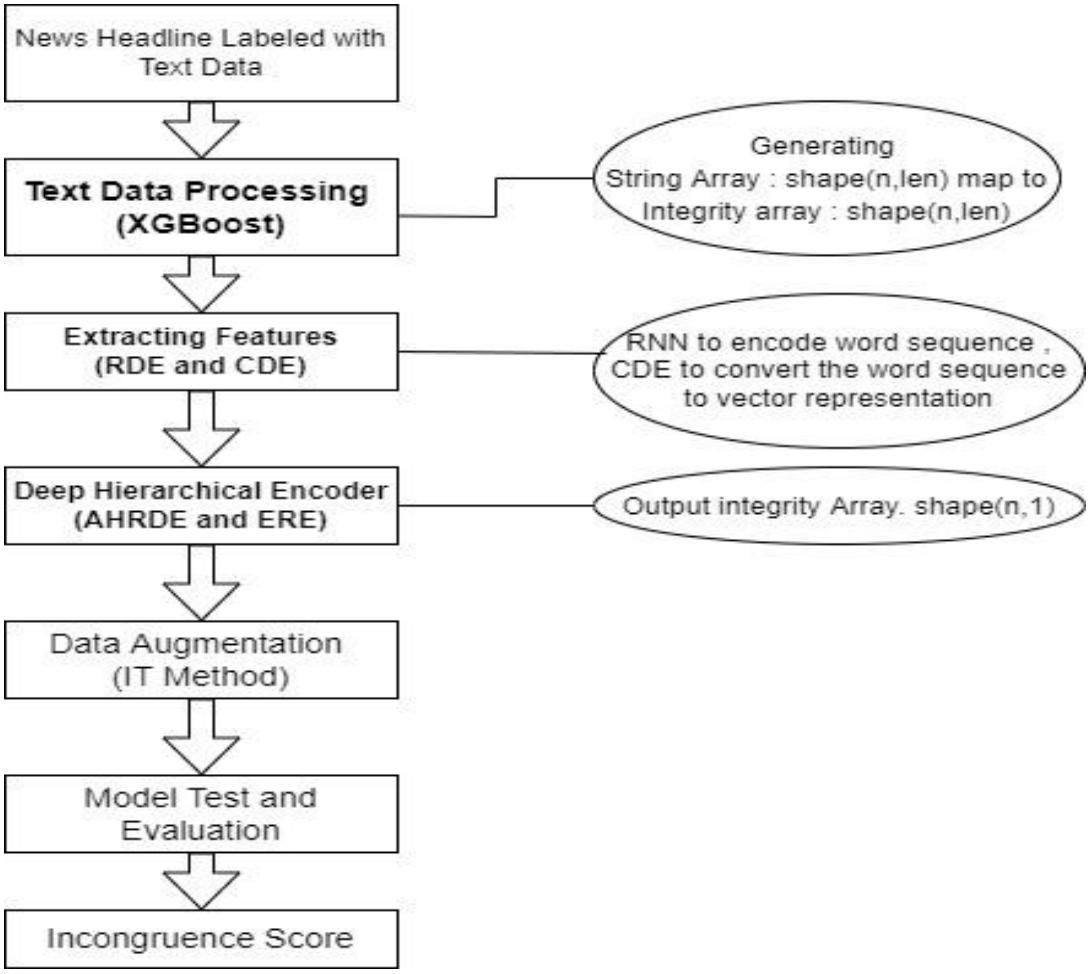

Figure 3.3: Flowchart showing the various modules of the Deep Hierarchical Encoder.