Attention-based Deep Learning Models for Text Classification and their Interpretability

by

Cansu Sen

A Dissertation

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Doctor of Philosophy

in

Computer Science

by

July, 2020

APPROVED:

Professor Elke A. Rundensteiner
Worcester Polytechnic Institute
Advisor

Professor Xiangnan Kong
Worcester Polytechnic Institute
Co-Advisor

Professor Carolina Ruiz
Worcester Polytechnic Institute
Committee Member

Dr. Preethi Raghavan
IBM Research
External Committee Member

Professor Craig Wills
Worcester Polytechnic Institute
Head of Department

Acknowledgements

I would like to express my gratitude to my advisor Professor Elke Rundensteiner, who cared so much about my work, for guiding and encouraging me through my Ph.D. studies. I appreciate her endless generosity with her time and attention, and helping me through many moments of uncertainty. I would like to thank her for the continuous support she showed, all of the opportunities she has created for me, and her patience.

I have been fortunate to have Professor Xiangnan Kong as my co-advisor. I am indebted for his time and invaluable guidance while following new research directions for this dissertation. Thank you, my committee members, Professor Carolina Ruiz, and Dr. Preethi Raghavan for valuable feedback and discussion at many stages of developing this dissertation. I appreciate your encouragement and input, which pushed this work forward in multiple ways.

Special thanks go to my close collaborator, Tom Hartvigsen, for his immense help, discussions, brainstorming, and valuable feedback over the years. I thank and acknowledge my collaborators at WPI: Jidapa Thadajarassiri, Dongyu Zhang, Biao Yin, Dr. Kajal Claypool, and Erin Teeple. A big shoutout to all the members of DSRG for friendship over the years, and to my WiDS co-ambassadors Caitlin Kuhlman and Melanie Jutras.

I thank the Computer Science Department and Data Science Program at WPI for granting me Arvid Anderson Fellowship and Teaching Assistantship throughout my Ph.D. My dissertation research was also partially funded by the National Science Foundation through grant IIS-1815866. I also thank the University of Massachusetts Medical School and Dr. Jomol Mathew for supporting my Ph.D. studies.

Finally, I couldn't have done this without loving family and friends who have ridden the highs and lows of these years with me. I am forever grateful to my understanding husband Mert Demirer, who endured this long process with me, always offering support and love. His advice and enthusiasm helped me do my research with greater motivation. Thank you, Mert, for being there the many times I needed emotional support. Lastly, I would like to thank my parents for encouraging and nurturing my curiosity all my life and always supporting me.

Cansu Sen

July 2020

ABSTRACT

Document series created over time is a prevalent data type in many domains, from healthcare to social media. A representative example of this type of data is clinical notes of a patient taken across time throughout the patient's hospital stay. These clinical notes correspond to hierarchical sequences of words and documents accompanied by the time information and other external attributes at each level of the hierarchy. Here, while the primary source of information is the nested word and document sequences, relevant meta-data, such as the creation time of documents or the patient's age, are also vital for accurately modeling these clinical notes.

In the first half of my dissertation, I design attention-based neural network models for modeling document series accompanied by time information. More particularly, I first focus on classifying hierarchical attributed sequences where categorical information is associated with different levels of the document hierarchy. To handle this, I propose HAC-RNN, composing of multiple Recurrent Neural Network (RNN) layers and an attributed hierarchical attention mechanism where each attention layer is conditioned on the external attributes. In HAC-RNN architecture, RNNs and attention layers are stacked hierarchically to account for the order of both words and documents. While the bottom layer of HAC-RNN is responsible for contextual summarization of the document content, the top layer considers the entire timeline and learns to concentrate on only the most relevant documents.

Second, with the observation that not just the sequential order, but the exact time-stamps at which documents are generated contains critical predictive power, I design a time-informed dual attention mechanism, TEND-LSTM. When classifying document series, TEND-LSTM learns how much attention to put on each document based on the content and time of creation independently. Then it learns to combine these two to generate final attention scores.

While these attention-based architectures are successful in classifying documents, it is not trivial to show that these models also achieve interpretability. Frequently,

to incorporate interpretability and account for long-term dependencies, attention-based neural architectures (e.g., attention paired with Recurrent Neural Networks, Transformer-based Networks) are used for modeling textual data. Despite extensive use, "correctness" and "interpretability" of the implicitly-learned attention weights have only been assessed qualitatively by visualizing a few hand-selected examples. Yet, designing interpretable models is of utmost importance in many domains. With this motivation, in the second half of my dissertation, I focus on dissecting and increasing the interpretability of attention-based neural networks.

To this end, I first investigate whether the common claim that attention mechanism increases model interpretability is correct by assessing how human-like are the explanations generated by a variety of attention mechanisms. I design a user study to collect "human attention maps" through crowd-sourcing for the publicly available Yelp Restaurant Review dataset. These human-attention maps are then utilized to quantitatively measure the similarity of human and machine-generated attention via novel similarity metrics specifically designed for this task.

Finally, extending the conclusions made from attention-similarity and using the collected human attention maps, I design a human-guided attention mechanism. The proposed human-guided attention mechanism learns to combine machine-inferred attention with human intuition to achieve improved classification performance and offer human-like reasonings for model predictions concurrently.

Publications

I provide a list of publications I produced during my Ph.D. studies at WPI.

1. **C. Sen**, T. Hartvigsen, E. Rundensteiner, K. Claypool, "CREST - Risk Prediction for Clostridium Difficile Infection Using Multimodal Data Mining", In Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD), pp. 52-63. Springer, Cham. 2017.

2. T. Hartvigsen, **C. Sen**, S. Brownell, E. Teeple, X. Kong, E. Rundensteiner, "Early Prediction of MRSA Infections Using Electronic Health Records", In International Conference on Health Informatics, (HEALTHINF), pp. 156-167. 2018. (Shortlisted for best student paper award)

3. T. Hartvigsen, **C. Sen**, E. Rundensteiner, "Detecting MRSA Infections by Fusing Structured and Unstructured Electronic Health Record Data", In International Joint Conference on Biomedical Engineering Systems and Technologies, pp. 399-419. Springer, Cham. 2018.

4. J. Friend, A. Hauck, S. Kurada, **C. Sen**, T. Hartvigsen, E. Rundensteiner, "Handling Missing Values in Multivariate Time Series Classification", IEEE MIT Undergraduate Research Technology Conference (URTC), 2018.

5. T. Hartvigsen, **C. Sen**, X. Kong, E. Rundensteiner, "Adaptive-Halting Policy Network for Early Classification", In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 101-110. 2019.

6. J. Thadajarassiri, **C. Sen**, T. Hartvigsen, X. Kong, E. Rundensteiner, "Comparing General and Locally-Learned WordEmbeddings for Clinical Text Mining", In 2019 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI), pp. 1-4. IEEE. 2019.

7. **C. Sen**, T. Hartvigsen, X. Kong, E. Rundensteiner, "Patient-level Classification on Clinical Note Sequences Guided by Attributed Hierarchical Attention", In 2019 IEEE International Conference on Big Data (Big Data), pp. 930-939. IEEE, 2019.

8. **C. Sen**, T. Hartvigsen, X. Kong, E. Rundensteiner, "Learning Temporal Relevance in Longitudinal Medical Notes", In 2019 IEEE International Conference on Big Data (Big Data), pp. 2474-2483. IEEE, 2019.

9. S.Wunnava, X. Qin, T. Kakar, **C. Sen**, E. Rundensteiner, X. Kong, "Adverse Drug Event Detection from Electronic Health Records Using Hierarchical Recurrent Neural Networks with Dual-Level Embeddings", Drug safety 42, no. 1 (2019): 113-122. 2019.

10. D. Zhang, J. Thadajarassiri, **C. Sen**, E. Rundensteiner, "Time-Aware Transformer-based Network for Clinical Note Series Prediction", In Machine Learning for Healthcare, 2020.

11. **C. Sen**, T. Hartvigsen, B. Yin, X. Kong, E. Rundensteiner, "Human Attention Maps for Text Classification: Do Humans and Neural Networks Focus on the Same Words?", In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 4596-4608. 2020.

12. T. Hartvigsen, **C. Sen**, X. Kong, E. Rundensteiner, "Recurrent Halting Chain for Early Multi-label Classification", In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2020.

13. T. Hartvigsen, **C. Sen**, X. Kong, E. Rundensteiner, "Learning to Selectively Update State Neurons in Recurrent Networks", In Proceedings of 29th ACM International Conference on Information and Knowledge Management (CIKM). 2020.

14. E. Teeple, T. Hartvigsen, **C. Sen**, K. Claypool, E. Rundensteiner, "Clinical Performance Evaluation of a Machine Learning System for Predicting Hospital-Acquired Clostridium Difficile Infection", In International Conference on Health Informatics, (HEALTHINF), pp. 656-663. 2020.

15. J. Thadajarassiri, **C. Sen**, T. Hartvigsen, X. Kong, E. Rundensteiner, "Similarity-Preserving Meta-Embedding", In Submission.

16. **C. Sen**, T. Hartvigsen, D. Zhang, J. Thadajarassiri, X. Kong, E. Rundensteiner, "Explainable Document Classification with Human-guided Attention", In Submission.

Among these publications, 2, 3, 4, 9, 10, 14 are the result of my collaboration with several Ph.D students including Thomas Hartvigsen, Erin Teeple, and Susmitha Wunnava. 6 and 15 are the result of my collaboration with Jida Thadajarassiri. 5, 12, 13 are the result of my collaboration with Thomas Hartvigsen.

Finally, 7, 8, 11 and 16 are the result of my research and they are discussed in this dissertation.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Deep Learning for Text Classification

With the advent of deep learning, the ways we process and utilize textual data for machine learning tasks has changed immensely. Especially with the developments in Recurrent Neural Networks (RNN) [29], a family of neural networks for processing sequential data, and word embedding tools [67, 73], these algorithms have been firmly established as state-of-the-art for problems spanning natural language processing, language modeling, machine translation, image captioning, handwriting recognition any many more [4, 47, 59, 101, 106]. RNNs capture nonlinear dynamics by learning a lossy summary of a future state using the past states. Some sequence models, such as Markov models, conditional random fields, and Kalman filters, deal with sequential data but are ill-equipped to learn long-range dependencies [33]. Other models require domain knowledge or feature engineering, offering less chance for serendipitous discovery. In contrast, neural networks learn representations and can discover unforeseen structure.

Despite their success, RNNs struggle when the input sequence is long [72]. Even Long Short term Memory (LSTM) Networks, a variant of RNN specifically designed for handling long sequences, are known not to work well when the input sequence becomes long [36].

Neural Attention models are proposed to solve some of the plights which RNNs struggle to handle [5]. Computational models of attention map a query and a set of key-value pairs to an output, where the query, keys, values, and output are all

vectors. The output is then computed as a weighted sum of the values, where the weight assigned to each value is generated by a compatibility function of the query with the corresponding key.

Attention mechanisms have become an integral part of compelling sequence modeling for two main reasons:

1) They exhibit better handling of long sequences by allowing modeling of dependencies without regard to their distance in the input or output sequences [81, 105].

2) They provide a weighted sum of model inputs; thus, they are believed to add transparency into black-box deep learning models [16, 93, 114].

However, there are two significant problems with these models being used for real-life applications.

1.2 Problem One: Real Data is More Complex Than Just a "Sequence of Words"

Most work for employing RNNs enhanced with attention is centered around classifying short text and a single document [62, 68, 94, 102, 107, 114]. However, many of the real-life text resources bear more complex characteristics and contain more abundant information.

For example, *document series* created over time is a prevalent data type in many domains. Longitudinal clinical notes of a patient, tweets/social media posts by a single user created over time, or conversations with a chat-bot can be regarded as an example of this data type. Document series naturally have a nested sequential structure. Namely, they are a sequence of documents created over time, and each document itself consists of a sequence of words.

Further, documents or document series are frequently accompanied by non-sequential meta-information at multiple levels. For example, the creation time of a document is valuable information at the document level. On the other hand, the demographic profile of the author of the document is user-level information which does not change at the document level. While the primary source of information is the nested word and document sequences, these external attributes are also vital for accurate modeling of this complex data type.

When used off-the-shelf, attention-based deep learning models cannot capture complex dependencies in document series.

1.2.1 Motivating Example: Clinical Notes of a Patient

Medical facilities across the United States have universally adopted Electronic Health Records (EHR) systems as a result of the Health Information Technology for Economic and Clinical Health (HITECH) Act[1] and the Centers for Medicare and Medicaid EHR Incentive Programs[2]. The widespread digitalization of health records presents a unique opportunity for health care innovation by using these sources of information with machine learning models [42, 83]. It is evident that there are signals embedded in these complex patient data that could indicate many medically important conditions.

EHR data is often multi-modal and heterogeneous, and consists of both structured (e.g., numerical sensor data) and unstructured (e.g., text) information. The structured part of EHR is straightforward in that meaningful information can be easily extracted from it. However, unstructured data in the form of free-hand notes recorded by clinicians make up a large portion of EHR databases and are more challenging to mine. Such text data, being a popular form of data entry for medical staff, often includes the intuition behind actions of experts, shown in general to be potentially rich with information [31, 89].

An example of a clinical note for a critical care patient is presented in Figure 1.1. This example is taken from the publicly available MIMIC EHR dataset [46]. Typically, a clinical note contains all medical events and facts about the patient, such as demographics (age:77), diagnosis (abdominal pain), procedures applied (chest AP portable single view). In addition, a clinical note may contain expert insight and intuition about the patient that cannot be found elsewhere in the EHR (e.g., "Mild cardiac enlargement is probably present"). Owing to this information-density, there is a great potential for developing deep learning models utilizing clinical notes for building clinical decision support systems to help to transform

[1] https://www.healthit.gov/topic/laws-regulation-and-policy/health-it-legislation

[2] https://www.cms.gov/Regulations-and-Guidance/Legislation/EHRIncentivePrograms/index.html?redirect=/EhrIncentivePrograms/

```
 [**3474-2-23**] 10:33 PM
 CHEST (PORTABLE AP); -77 BY DIFFERENT PHYSICIAN
[**Name Initial (PRE) 24**] # [**Clip Number (Radiology) 85536**]
 Reason: acute chest pain, tachypnea
 Admitting Diagnosis: ABDOMINAL PAIN
_____________________________________________________________________________
UNDERLYING MEDICAL CONDITION:
  77 year old man with CHF, PNA, hypoxia s/p open appy and decompression of
 [**Last Name (un) 10326**].
REASON FOR THIS EXAMINATION:
 acute chest pain, tachypnea
_____________________________________________________________________________
                                FINAL REPORT
TYPE OF EXAMINATION:  Chest AP portable single view.

INDICATION:  CHF and pneumonia, status post appendectomy and decompression.
Acute chest pain and tachypnea.

FINDINGS:  An AP single chest view has been obtained with the patient in
supine position and comparison is made with a similar previous examination
obtained seven hours earlier during the same day.  Films are of different
image quality as the preceding portable chest examination apparently was taken
without grids.  Mild cardiac enlargement is probably present and there is a
certain degree of perivascular haze similar as described on the preceding
examination.  There is however no evidence of overt interstitial or alveolar
edema and the lateral pleural sinus on the right side remains free.  The
portable film does not cover the entire chest so that the left lateralchest
base is cut off.  There is no evidence of any pneumothorax.

IMPRESSION:  No significant interval change over the last seven hours.
```

Figure 1.1: Example of a clinical note for a critical care patient. This example is taken from the publicly available MIMIC EHR dataset [46] and private information is deducted. Typically, a clinical note contains all medical events and facts about the patient such as demographics, medications given, procedures applied, etc. In addition, it contains expert insights about the patient that cannot be found elsewhere in the EHR (e.g., "Mild cardiac enlargement is probably present")

the Healthcare industry.

1.3 Problem Two: Can We Call Attention Interpretable?

The most significant limitation of the deep learning models is their lack of explainability. As neural networks become more and more complex, they also become more *black-box* [68, 88]. Yet in contrast, the interpretability of a machine learning model gains immense importance when it is used for real-life applications.

Interpretability is an indispensable feature in many domains including healthcare, self-driving cars, and criminal applications, where there is potential effect on human life.

An attention function essentially computes a probability distribution over the input space. This probability distribution, thus, can be interpreted as the "feature importances". Since the learned attention scores inform us about which part of the input is weighted more heavily when making a classification decision, researchers tend to think that these scores add transparency into black-box deep learning models.

But, what does it mean if a model puts a higher weight into some parts of the input? Can we claim that a reason for a model reaching a classification decision would be similar to human justification? Can we claim that attention adds any interpretability?

The definition of interpretability is crucial in answering these questions. Even though there is no consensus on a single definition of interpretability, we can intuitively conclude that an ordinary person should understand why a machine learning model produces any classification decision. For example, if an ML system is being used to detect likely-criminals, the machine learning model must explain why it predicts a particular individual potential-guilty. Only this way, humans can judge if they can trust auto-generated decisions.

Whether attention can serve this purpose cannot be understood only through qualitative analysis of the machine-learned attention scores, as usually done in the literature. The claim that the attention weights correspond to *human-understandable* rationales for model predictions requires further work and analysis.

1.4 State-of-the-Art

1.4.1 Clinical Note Classification

Clinical notes of a patient typically make up a series of documents created over time. An important task is to use these document series to make predictions about the patient's future state. A large body of literature employs non-neural network based architectures to tackle this problem. Many researchers utilize bag-

of-words representations followed by linear classification methods such as SVM to classify clinical notes [10, 44, 76]. In [31] and [32], authors examine latent variable models, namely LDA, to decompose free-text notes into features for mortality prediction. They divide the hospital stay of a patient into time windows and then extract features from aggregated notes within each time window. LDA and topic modeling techniques are used in other studies including for intervention prediction [100] and for readmission prediction [89]. In [11], noun-based, term-based, and topic-based features are extracted from clinical notes for named-entities through medical dictionaries such as SNOMED [95]. More recently, [27] embraces two deep learning approaches for learning representations from clinical notes. The first approach uses GloVe [73] to learn low-dimensional dense embeddings of clinical terms. Patient-level representations are derived by aggregating the embeddings. The second approach uses an RNN with bag-of-words representations of a sequence of notes. Clinical notes of a patient exhibit a nested sequential data structure: the order of words corresponds to the semantic axis, while the order of documents represents the time axis. Aforementioned state-of-the-art methods for patient-level classification of clinical notes ignore information from one or even both of these two axes.

1.4.2 Time-aware Models

A key characteristic of the clinical note series is that they include the time information at which they were created. This time information may be of equal importance to the content of the note when estimating a patient's future state. Time-aware recurrent networks or attention mechanisms concentrate on incorporating time information into the model decisions while classifying inputs. With this aim, a line of research proposes ways of modifying the RNN cell to account for time. For example, [118] uses a time decay term in the update gate in GRU to find a trade-off between the previous hidden state and the candidate hidden state. [74] extends the forget gate of the standard LSTM unit to a logarithmic or cubic decay function of time intervals between two time stamps. [12] applies a time decay function to the previous hidden state in Gated Recurrent Unit (GRU) before calculating the new hidden state. [9] first decomposes memory cell in LSTM into long-term memory and short-term memory then applies time decay to discount the short

term memory and finally calculates the new memory by combining the long-term memory and a discounted short-term memory. The main goal in these papers is to handle missing values in time series data; hence, they attempt to discount the effect of an observation if more time passed, which is not always true. In addition, modifying RNN units limits the interpretability of the resulting models, since RNN units are usually treated as black boxes.

Recently, simple time-attention mechanisms have been proposed within the spoken language understanding domain [14, 96, 97]. In these works, either a hand-picked fixed function of time [14] or a parameterized time-decay function [96] serve as the attention weights. [6] uses a disease progression function to control how much information flows into RNN at each time step. The input to these time functions corresponds to a scalar representation of time, namely the time difference between instances. It is overlooked that other representations of time have the potential to be even more informative for certain tasks and domains – which would not be known prior to learning a model.

1.4.3 Interpretability of Neural Attention Models

Despite their success, deep-learning models suffer from an explainability problem. However, in many domains including health-care, explainable models are of vital importance. Various methods have been proposed to make neural network models more interpretable. "Rationale-based" methods are an example of this effort [7, 57]. Here, the goal is to train a classification model and at the same time produce binary "rationales" to serve as human-like explanations to model predictions. As an alternative approach, a large body of work uses attention mechanism to bring interpretability to model predictions; however, they only assess the quality of produced attention maps qualitatively, by visualizations of a few hand-selected instances [15, 60, 119]

In [21], authors conduct the first qualitative assessment of computational attention mechanisms for Visual Question answering (VQA) task. In this work, the authors collect and publish VQA-HAT human attention dataset and using this dataset they measure the similarity of human and machine attention within the context of VQA. VQA-HAT dataset has provided researches with the opportunity of supervising the attention mechanism. A similar dataset and a quantitative eval-

uation is necessary but still missing for the Natural Language Processing domain. Existence of such datasets allows both quantitative assessments of computational attention mechanisms and training supervised attention mechanism as seen in [60].

1.4.4 Supervised Attention

Current state-of-the-art attention mechanisms learn to produce attention scores for the input space in an unsupervised manner. However, models without attention supervision may generate inaccurate attention maps. Inaccurate attention maps degrade the interpretability of model, as well as may result in incorrect predictions. Recently, researchers have started searching for ways of supervising the attention mechanism. For example, Chen et al. [13] propose using an auxiliary learning task to generate weak supervision for the attention mechanism for action localization in video frames. Liu et al. [60] encourage the model to "attend to" the regions of the image that humans pay more attention using the previously published human attention datasets for image captioning task.

Supervising the attention is a relatively new direction in NLP domain. Even though some works exist, most of them base their methods on pre-defined word lists deemed more important or relevant than the others to guide the attention mechanism. In [63], they use a strategy where "argument words" should acquire more attention than other words. For this, they define a list of words to receive "gold attention" in a binary form, then employ them as supervision to train the attention mechanism. Similar approaches in a hierarchical manner are proposed in [70, 117]. In [61] guidance from conventional alignment models are used to supervise the attention mechanism for Neural Machine Translation Task (NMT). However, a context-dependant "human-supervision" has not been proposed thus far possibly due to the lack of such human-attention datasets.

1.5 Research Challenges

Patient-level Classification on Clinical Note Sequences. Utilizing clinical note sequences to produce a patient level classification requires the following challenges to be addressed:

- *Nested Sequential Structure*: Patient records contain time series of notes, each of which consists of a sequence of words. Hence they cannot be directly fed into an RNN without losing crucial sequential information inherent to at least one layer. Moreover, *sequential* information is accompanied by external attributes at multiple layers. For example, every document is composed of not only a *sequence* of words but also a creation time, which is *static* across the sequence.

- *Long Term Dependencies*: Every note is typically a long document, composed of many hundreds of words. Further, every patient has a fairly high number of clinical notes taken over time. In practice, state-of-the-art text mining models continue to struggle to capture long-term dependencies. Even methods designed for this challenge are empirically difficult to optimize.

- *Unknown Temporal Importance*: The exact time when a clinical note is created may be of equal importance to the content of the note when estimating a patient's future state. For example, if a patient develops fever 2 hours later than having a belly pain, this may indicate a certain infection. On the other hand, if these two events happen days apart from each other, then these medical events do not necessarily indicate to an infection risk. The true function relating *when notes are recorded* and *clinical outcomes* is naturally unsupervised. Additionally, this relationship may change per task, so defining one fixed temporal importance function is too rigid. Previous works input a single time representation to their model (time difference between two consecutive time steps) to represent temporal importance; while other time functions may much better match the underlying relationship between the time occurrence of instances and how much attention they should receive.

- *Balancing Multiple Sources of Attention*: As described above, there are multiple factors to pay attention when making patient-level predictions on sequences of clinical notes. Thus, the goal is not only to decide how much attention to put on each document based on their content but also based on the time at which they were created. Instead of rigid combinations of these two attention sources, a balance among them must be *learned* as the correct balance is task-dependent.

Interpretability of Neural Attention Models. In the clinical domain, using a model that matches the complexity of the data providing an optimized performance is not sufficient for an ideal clinical decision support system. Such a system must also tackle the following challenges.

- *Required Interpretability*: Neural network-based text classification tends to be "black-box" in the sense that the models are challenging to interpret. However, in the clinical setting, interpretability is paramount to provide clinicians insights into how models choose particular predictions.

- *Unknown Behaviour of Attention Mechanism.* In addition to achieving significant performance gains, attention models are attractive as they are often used as a proxy for human-interpretable rationales for model decisions [7, 114]. The implicit assumption is that machine-generated attention *mimics* human behavior. However, machine-generated attention thus far has not been evaluated quantitatively to measure its similarity to human attention. Collecting and studying human-attention maps is imperative for a systematic quantitative assessment of machine-generated attention.

- *Quantifying the Similarity Between Human and Machine*: Numerical representations of human and machine attentions don't just denote which tokens are given higher importance, but also carry information about the underlying grammatical structure and linguistic construction. For example, if words deemed high-importance by human or neural network models tend to be adjectives, this information is embedded into the attention vectors. Therefore we must design a similarity metric capable of incorporating this diverse information, rather than just comparing two numerical vectors. In addition, we must address that human attention is a subjective concept. That is, no one human's attention can be regarded as the ground-truth for attention.

- *Lack of Supervision for Attention Mechanism*: Models without attention supervision may generate inaccurate attention maps, which in turn impairs the interpretability of model. This is due to true attention distribution is not known prior to model training. Collection of human-attention maps enable training of human-guided attention mechanisms.

1.6 List of Tasks Covered in this Dissertation

In this dissertation, I study four tasks involving novel attention-based architectures for modeling documents and specifically time series of documents with an emphasis on clinical notes. Each of these four tasks concentrates on a different problem and proposes a method addressing it.

The tasks are summarized as follows:

1. **Attributed Hierarchical Attention.** First, I focus on classifying sequences of clinical notes which correspond to hierarchical attributed sequences (HAS). HAS data type typically includes sequential information that is hierarchically organized, and categorical information is associated with different levels of the hierarchy. I propose the HAC-RNN deep neural architecture, composed of multiple RNN layers and an attributed hierarchical attention mechanism where each attention layer is conditioned on the external attributes. In HAC-RNN architecture, RNNs and attention layers are stacked hierarchically to account for the order of both words and documents. While the bottom layer of HAC-RNN is responsible for contextual summarization of the document content, the top layer considers the entire timeline and learns to concentrate on only the most relevant documents.

2. **Time-aware Dual Attention.** Second, with the observation that not just the sequential order, but the exact time-stamps at which documents are generated contains critical predictive power, I design a time-aware attention model. This novel attention mechanism composed of dual-attention blocks based on a rich diversity of time representations. I then pair this mechanism with an LSTM, resulting in our proposed time aware recurrent network TEND-LSTM. TEND-LSTM learns an integrated set of attention weights, with the first attention based on the content of the clinical notes and the second based on when the notes were taken. Together, they are combined using a deep-attention network layer. The proposed dual attention mechanism not only learns a function of time incorporating different aspects of the temporal nature of note instances but also automatically finds a balance between how much attention to put on content versus time.

3. **Human vs Machine Attention.** Third, I further focus on the interpretability of attention and analyze whether machine-learned attention is similar to human attention. For this, I conduct the first quantitative assessment of human versus computational attention mechanisms for the text classification task. To achieve this, I design and conduct a large-scale crowd-sourcing study to collect human attention maps that encode the parts of a text that humans focus on when conducting text classification. Based on this new resource of human attention dataset for text classification, YELP-HAT, collected on the publicly available YELP dataset, I perform a quantitative comparative analysis of machine attention maps created by deep learning models and human attention maps. Our analysis offers insights into the relationships between human versus machine attention maps along three dimensions: overlap in word selections, distribution over lexical categories, and context-dependency of sentiment polarity.

4. **Human-Guided Attention.** Finally, extending the conclusions made from attention-similarity and using the collected human attention maps, I propose a novel explainable attention mechanism, called Human-Guided attention, HUG, and a complementary learning scheme that facilitates human guidance on attention training. The HUG attention mechanism can be paired with any sequential deep learning architecture such as RNN or BERT. The HUG learning scheme integrates multiple objectives at different granularities of words and documents to learn a human-guided attention distribution over words, while also achieving a document-level classification task. Unlike in traditional attention mechanisms that learn the attention scores unsupervised, our model employs a direct learning paradigm that penalizes the model as its attention scores differ from human attention.

1.7 Dissertation Organization

This dissertation is organized as follows. I first describe some background material that will be frequently used throughout the chapters in Section 2. I introduce the Attributed Hierarchical Attention model in Section 3 and Time-Enhanced Dual Attention model in Section 4. Section 5 describes our data collection

study for human attention maps, and our analysis for comparing human attention to machine-generated attention. In Section 6, I propose a human guided attention model that improves the classification task accuracy and model interpretability concurrently. I describe related work in Section 7. Finally, the conclusion and future work are presented in Sections 8 and 9.

Chapter 2

Background: Neural Methods for Mining Textual Data

2.1 Recurrent Neural Networks

Recurrent Neural Network (RNN) and its variations has been extensively used in diverse tasks on sequential data, such as handwriting recognition [35], speech recognition [34] and document classification [114]. Similar to other neural networks, RNN utilizes the backpropagation algorithm to calculate gradients that could be used to fit the model parameters.

Unlike other neural networks where each item in the input sequence is processed independently, RNN assumes that there are temporal dependencies across time steps in each sequence [28]. To model these sequential dependencies, RNN shares model parameters across time steps when processing each input. That is, RNN not only takes the input at the current time step into account but also all other time steps processed previously. The computational graph of RNN is illustrated in Figure 2.1.

A recurrent network maps an input sequence of x to a corresponding sequence of output values o. A loss of L measures how far each o is from the corresponding training target y. When using softmax outputs, we assume o is the unnormalized log probabilities. The loss L internally computes $\hat{y} = \text{softmax}(o)$ and compares this to the target y. The RNN has input to hidden connections parameterized by a weight matrix U, hidden-to-hidden recurrent connections parameterized by a

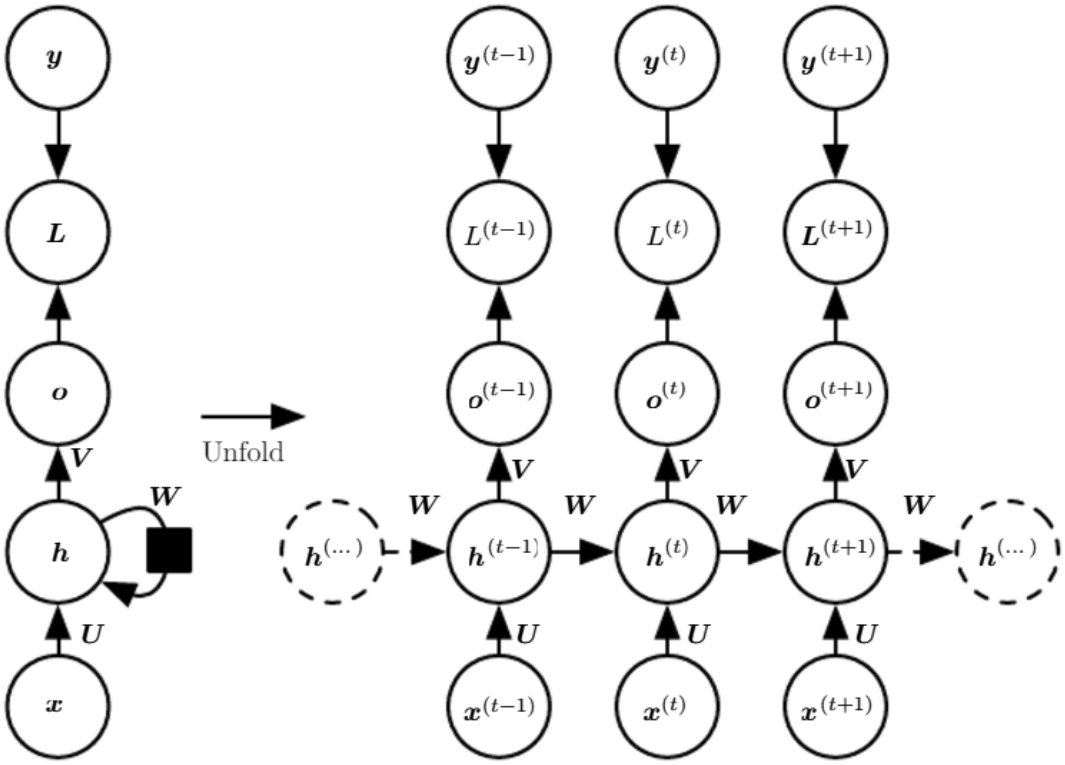


Figure 2.1: Schematic representation of an RNN [33].

weight matrix W, and hidden-to-output connections parameterized by a weight matrix V.

For each sequence input, the sequential information is preserved in RNN's hidden state, which is continuously updated as long as the RNN cascades forward to process future time steps. For this reason, the hidden state is often referred to as the "memory" of the RNN in the literature [39]. Given the input x_t at time step t, the respective RNN's hidden state is often defined as:

$$h_t = \sigma(Wx_t + Uh_{t-1}) \tag{2.1}$$

where the respective weight matrices for "input-to-hidden" and "hidden-to-hidden" are denoted as W and U. The activation function could be either a logistic sigmoid function or a hyperbolic tangent function, selection of which depending on specific tasks. The hidden state h_t can be exploited for various tasks, such as next word prediction and sequence classification.

In this dissertation, many proposed machine learning models are based on RNNs and its variations, LSTM and GRU. Next, I describe these cell structures.

2.1.1 Long Short-Term Memory (LSTM)

Vanilla RNN suffers from the problem of exploding and vanishing gradient during the training, where the gradient value becomes too large or small. This may result in the network becomes untrainable. Initially proposed for addressing this issue, LSTM [39] is one variation and expansion of the vanilla recurrent neural network. Similar to a Vanilla RNN with a chain of repeating cells, LSTM cells are also chained together to handle sequential inputs.

In addition to the gradient issues, Vanilla RNN also suffers from handling long sequences, where the past information degrades over time, even though being designed to take into account all prior information. Through a more sophisticated design by using a memory cell and three gates (i.e., input gate, output gate, and forget gate), LSTM is capable of *remembering* values over long time intervals. The term "long short-term memory" refers to the fact that LSTM is capable of preserving short-term memory for a long time. Specifically, compared to the simple design of Vanilla RNN, where only one sigmoidal function is used, there are five

functions used in an LSTM cell.

Using input x at step t and previous state h_{t-1}, gates are computed as

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2.2}$$

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{2.3}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{2.4}$$

where W_i, W_f, W_o and b_i, b_f, b_o are parameters shared across time and σ is the sigmoid function.

Then a cell state is computed as

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{2.5}$$

where W_c and b_c are parameters shared across time.

Finally, the hidden state h_t is

$$h_t = o_t \odot \tanh(c_t) \tag{2.6}$$

LSTM has been used in many real-world applications involving sequential inputs. For example, Google and Amazon uses LSTM for speech recognition [49, 110]. LSTM has also achieved state-of-he-art performance in machine translation [101], image captioning [47], hand writing recognition [35] and question answering [103].

2.1.2 GRU

GRU was initially proposed for machine translation task in [18]. Both GRU and LSTM have the goal of tracking long-term dependencies while mitigating the problem of exploding and vanishing gradient. Different from LSTM, where there are three gates and two internal states are used, GRU only have two gates (namely, the reset and update gates) and one internal state. A GRU is defined as:

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \tag{2.7}$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \tag{2.8}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tanh(w_h x_t + U_h(r_t \odot h_{t-1}) + b_h) \tag{2.9}$$

where z_t and r_t denote the update gate and reset gate, respectively. W_z, W_r, W_h, U_z, U_r, U_h, b_z, b_r, b_h are trainable model parameters. h_t is the internal state and it is also served as the output of a GRU cell.

There are two main differences between GRU and LSTM. First, the exposure of the memory in LSTM is controlled through the output gate, which is missing in the GRU model. That is, only part of the memory is exposed in LSTM compared to full memory exposure in GRU. Second, LSTM controls the amount of new information flowing into the model by using two gates, namely the input gate and the forget gate. The control of the information flow is through the forget gate, independently. Meanwhile, GRU controls the information from the previous activation when computing the new activation from the new input at the same time.

2.2 Attention Mechanism

Inspired by human attention, a recent trend in deep learning is to build computational models of attention [5]. An attention function can be described as mapping a query and a set of key-value pairs to an output, where each of the query, keys, values, and output are all vectors [105]. Through a compatibility function of the query with the corresponding key, weights are generated for each value. The output of the attention function is the weighted sum of the values. Attention mechanisms are now an integral component of many sequence modeling tasks. They enhance models with the ability to deal with long sequences.

2.2.1 Additive Attention for Sequence to Sequence Classification

This type of attention is introduced by [5] for machine translation task. Neural machine translation is a sequence to sequence classification task. The mail model is an encoder-decoder RNN, and the goal is to learn an alignment model between inputs and outputs through an attention function.

Assume that s_i is the hidden state of the decoder for time i and it is computed as:

$$s_i = f(s_{i-1}, y_{i-1}, c_i) \tag{2.10}$$

The context vector c_i is a function of the sequence of annotations $(h_1, \ldots, h_{T_x})$ to which an encoder maps the input sentence. Each annotation h_i represents the whole input sequence, while also focusing on the other parts surrounding the i-th word of the input sequence. The context vector c_i corresponds to weighted sum of the annotations h_i:

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j \tag{2.11}$$

The weight α_{ij} of each annotation h_j is computed by:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_k \exp(e_{ik})} \tag{2.12}$$

where $e_{ij} = a(s_{i-1}, h_j)$ is an alignment model. This alignment model represents how well the inputs around position j and the output at position i match. Attention function is modeled as a feedforward neural network, and it is jointly trained with all the other components of the model.

The probability α_{ij}, which is the attention score, shows the importance of the annotation h_j with respect to the previous hidden state s_{i-1} in deciding the next state s_i and generating y_i. Intuitively, this corresponds to a mechanism of attention in the decoder.

2.2.2 Additive Attention for Sequence Classification

This type of attention is a variant of the original attention mechanism proposed by [5], where it is adapted for the sequence classification task, such as document classification [114].

Additive attention for sequence classification often paired with a recurrent model. Assuming that Γ is a recurrence function and x_i is the embedded i-th token of T tokens in a sequence, additive attention is modeled as:

$$h_i = \Gamma(x_i, h_{i-1}), i \in [1, T] \tag{2.13}$$

$$u_i = \tanh(W h_i + b) \tag{2.14}$$

$$\alpha_i = \frac{\exp(u_i^\top u)}{\sum_t \exp(u_i^\top u)} \tag{2.15}$$

Here $h_i, i \in [1, T]$ are hidden representations, W, b, and u are trainable parameters, and $\alpha_i, i \in [1, T]$ are the attention scores for each input token x_i. A context vector c corresponds to the weighted average of the hidden representations of inputs with attention weights, denoted by:

$$c = \sum_j \alpha_j h_j \tag{2.16}$$

Through a softmax function, context vector c_i is then used for further classifying the input sequence.

Many models in this dissertation uses additive attention mechanism for the sequence classification task.

Attention mechanism can help for classification tasks when the input sequence is very long. For example, given a sequence of documents related to a patient, neither all terms in a document nor each document are equally relevant for predicting a task and determining the relevant sections involves modeling the interactions of the words, not just their presence or absence. Figure 2.2 shows an RNN architecture, both with and without attention. An RNN, while expected to create a summary of input space with one vector at the end of the sequence, can fail to achieve this. On the contrary, added attention mechanism helps adjusting focus on parts of the input, relieving the burden of remembering essential information across the

sequence.

2.3 Word Embeddings and Word2Vec

Word embeddings are prevalently used to transform words into numeric vectors. These vectors preserve relationships between words by considering the context in which each word appears. Learning word embeddings is time-consuming, especially when the source text is large. For this reason, many pre-trained word embeddings are published to help researchers bypass the learning process and commonly used as the baselines for text classification. However, vector representations can differ dramatically between embedding sources depending on both the size and content of the training corpora. Thus, locally-learned embedding is another appealing choice to use for clinical machine learning.

Unless stated otherwise, locally-learned word embeddings are used for the tasks in this dissertation proposal.

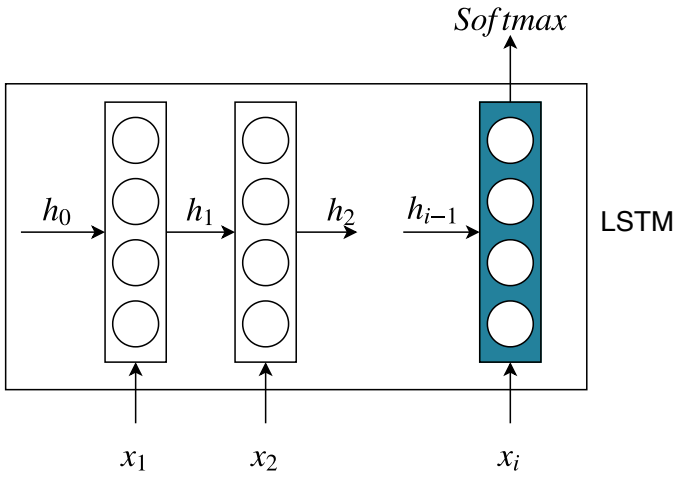
We learn word embeddings prior to end-to-end model training disjointly, using the skip-gram model of the "word2vec" algorithm [67].

Skip-Gram assumes that words appear in similar context are likely to have similar meanings. Thus, pairs of context-target words are created from an input corpus. The model predicts context words from a given target word, i.e., for a target word w_t and a given number of contextual-window-size c, Skip-Gram will predict the probability of c words before and c words after w_t. $\mathbf{x}_{w_t}$, a one-hot encoding of w_t, is fed into a 2-layer neural network. The final weight matrix of the 1^{st} layer is the desired word embedding.

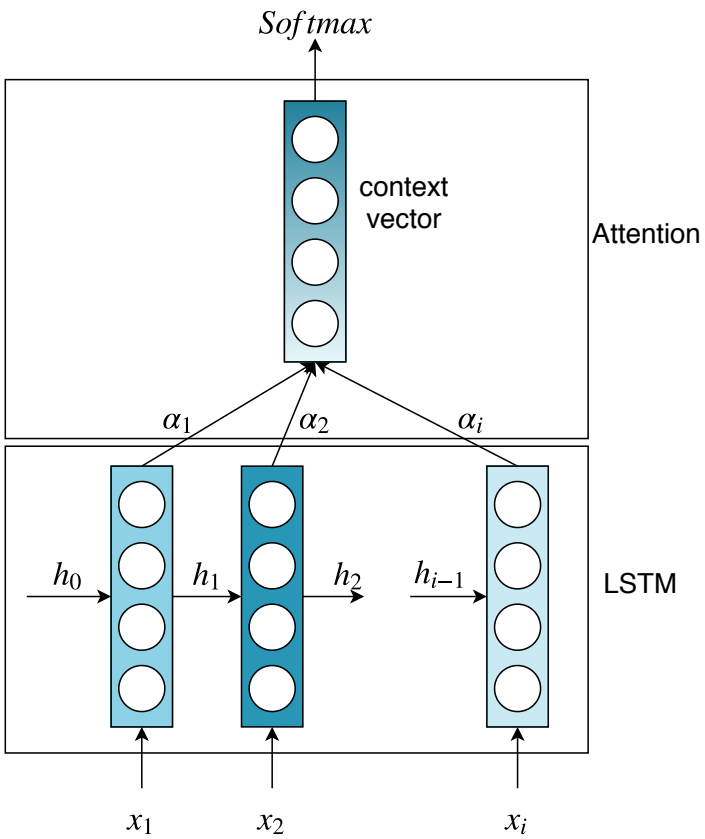
Let $\mathbf{w} = \{w_1, w_2, ..., w_T\}$ be the T training words and d be the number of hidden units. The size of weight matrices of 1^{st} layer ($\mathbf{W}$) and 2^{nd} layer ($\mathbf{W}'$) are $T \times d$ and $d \times T$ respectively. The t^{th} row of $\mathbf{W}$, which is the d-dimensional vector representation $\mathbf{v}_{w_t}$ of w_t, is copied to the hidden layer.

$$\mathbf{h}_t = \mathbf{x}_{w_t}^T \mathbf{W} \text{ and } \mathbf{u} = \mathbf{W}^T \mathbf{h}_t^T \tag{1}$$

where $u_j \in \mathbf{u}$ is the predicted score of word w_j being the context word. A softmax activation function is lastly applied to obtain the probability distribution



(a) LSTM



(b) LSTM with attention mechanism

Figure 2.2: Architectural comparison of LSTM, with and without the attention mechanism.

in the output layer:

$$p(w_j|w_t) = \hat{y}_j = \frac{exp(u_j)}{\sum_{j'=1}^{T} exp(u_{j'})} \tag{2}$$

where $\hat{y}_j$ is the probability of w_j being the context word.

Model is trained using the objective function:

$$\max \frac{1}{T} \sum_{t=1}^{T} \sum_{\substack{j=t-c \\ j \neq t}}^{t+c} \log p(w_j|w_t) \tag{3}$$

2.4 Transformer-Based Architectures

2.4.1 Transformer Architecture

Before the introduction of the transformer mechanism, Recurrent Neural Networks, in particular, long short-term memory [39] and gated recurrent [18] neural networks, have been established as state-of-the-art approaches in sequence modeling problems such as language modeling and machine translation. Attention mechanisms have become a fundamental part of sequence modeling as they often improve the overall performance of the model and allow for modeling dependencies without regard to their distance in the input or output sequences. Such attention mechanisms are almost always used in conjunction with a recurrent network [5, 20, 54, 98, 114].

Transformer [105] is a model architecture based only on attention mechanisms, as opposed to having recurrence functions. Transformer architecture is proposed for the machine translation task, and it is composed of encoder and decoder layers, both of which contain attention heads and fully-connected layers.

2.4.1.1 Encoder-Decoder Model

Transformers use an encoder-decoder architecture. The encoder first maps an input sequence of symbol representations $(x_1, \ldots, x_n$ to a sequence of continuous representations $z = (z_1, \ldots, z_n$. Given z, the decoder generates an output sequence $(y_1, \ldots, y_n$ of symbols one element at a time. At each step, the model is auto-regressive, consuming the previously generated symbols as additional input when generating the next.

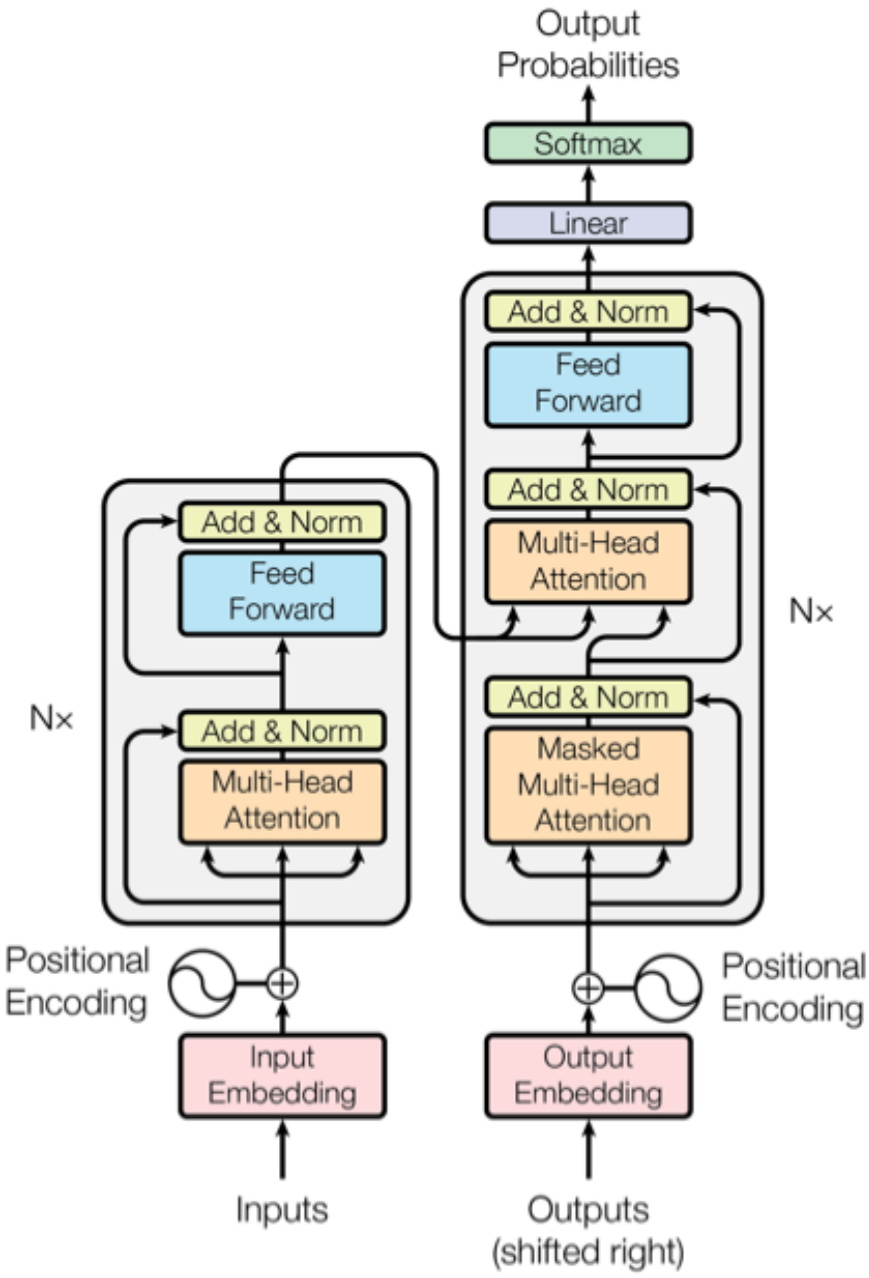


Figure 2.3: Encoder and decoder architecture of the Transformer model [105].

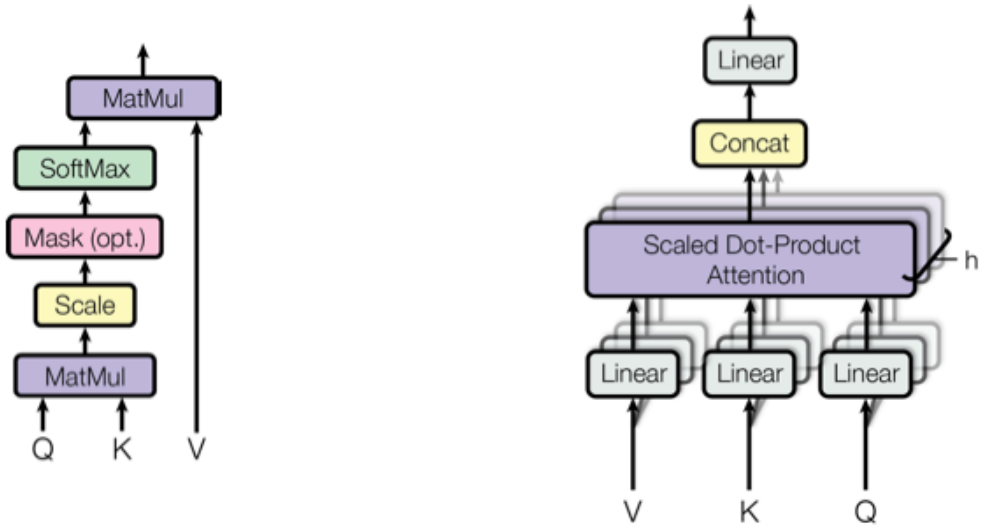


Figure 2.4: Left figure shows the details of scaled dot-product attention. Right figure depicts how multiple dot-product attention layers run in parallel [105].

The encoder subnet consists of N identical layers. Each encoder layer includes two sub-layers. The first sub-layer is a multi-head self-attention mechanism. The second sub-layer is a fully connected feed-forward neural network.

The decoder subnet has a similar stack of layers. In addition to the two sublayers used in the encoder, the decoder also includes a third sub-layer. This third layer is another attention layer between the output of the encoder layer and the decoder. The self-attention sub-layer in the decoder is modified to prevent positions from attending to subsequent positions. This is done to ensure that the predictions for position i can depend only on the known outputs at positions less than i.

The transformer architecture is depicted in Figure 2.3.

2.4.1.2 Scaled Dot Product Attention

Transformer architecture introduces a novel attention mechanism, referred to as Scaled Dot-Product Attention (Figure 2.4). In the scaled dot product attention, the inputs are keys and queries of size d_k and values of size d_v. The compute this attention, we first calculate the dot product between the query with all keys, and we divide it by $\sqrt{d_k}$. We then apply a softmax function to obtain the weights on the values. In practice, this process is vectorized, and the attention function is

computed on a set of queries simultaneously via the query matrix Q. The same operation is conducted on the key and values to build the matrices K and V. We compute the matrix of outputs as:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{2.17}$$

2.4.1.3 Comparison with RNN-based Architectures

Transformer and RNN-based architectures enhanced with attention mechanisms have a number of differences.

First, transformer architecture only uses self-attention and fully-connected layers. It does not use recurrent or convolutional layers at all, even though its input is sequential text data.

Second, as opposed to RNN-based architectures, Transformer allows for parallelization. This is one of the reasons why the Transformer is much faster. For translation tasks, the Transformer can be trained significantly faster than architectures based on recurrent or convolutional layers.

Third, the Transformer is uni-directional, whereas RNNs can be bidirectional. However, this shortcoming is addressed on the successive architectures starting from BERT [23].

Lastly, [55] compares the performance of the Transformer and RNN-based architectures for various neural machine translation (NMT) tasks. Based on experimental results, the Transformer approach delivers the best performing multilingual models, with more significant gains over corresponding bilingual models than observed with RNNs. Even though transformers only focused on NMT task, later transformer-based architectures show superior performance over RNN-based models (e.g., BERT-large [23] in named entity recognition, RoBERTa [64] in natural language inference)

Transformers drew immediate attention from the NLP community and had a huge impact on the NLP domain. They show superior performance on the GLUE Benchmark dataset and stayed on top of the charts in the Glue Leaderboard [1]. Many researchers then followed the lead of [105] and proposed alternative architectures that improve the performance. These architectures include BERT [23], XLNet

[1] https://gluebenchmark.com/leaderboard

[113], RoBERTa [64], ERNIE [99], and OpenAI GPT [80].

2.4.2 BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

GPT introduces minimal task-specific parameters and is trained on the downstream tasks by simply fine-tuning all pre-trained parameters. However, it uses a uni-directional language model (left-to-right), where every token can only attend to previous tokens in the self-attention layers of the Transformer, to learn general language representations. This restricts the power of pre-trained representations and limits the choice of architectures that can be used during pre-training. Such restrictions are sub-optimal for sentence-level tasks and could be harmful when fine-tuning on token-level tasks such as question answering, where it is crucial to incorporate context from both directions.

BERT: Bidirectional Encoder Representations from Transformers [23] addresses this problem, advancing the state-of-the-art for many NLP tasks. The key to BERT's success is a learning paradigm referred to as Masked language modeling (MLM). The masked language modeling task requires randomly masking some of the input tokens and then predicting the vocabulary id of the masked word based only on its context. As opposed to the left-to-right language model pre-training, as has done in previous architectures, the MLM objective enables the representation to combine the left and the right context. As a result, this allows us to pre-train a deep bidirectional Transformer. Another novelty BERT introduces is a second pre-training task, which is called "next sentence prediction".

Pre-training of BERT. We use two tasks for pre-training BERT as described below.

Task 1: Masked Language Modeling. Masked language modeling is one of the two tasks used for pre-training a BERT model. This task requires masking some percentage of the input tokens randomly. Then the model learns to predict those masked tokens. The final hidden vectors corresponding to the mask tokens are fed into a softmax function over the words in the vocabulary, similar to standard language modeling tasks. In BERT architecture, 15% of input tokens in each sequence selected randomly are masked.

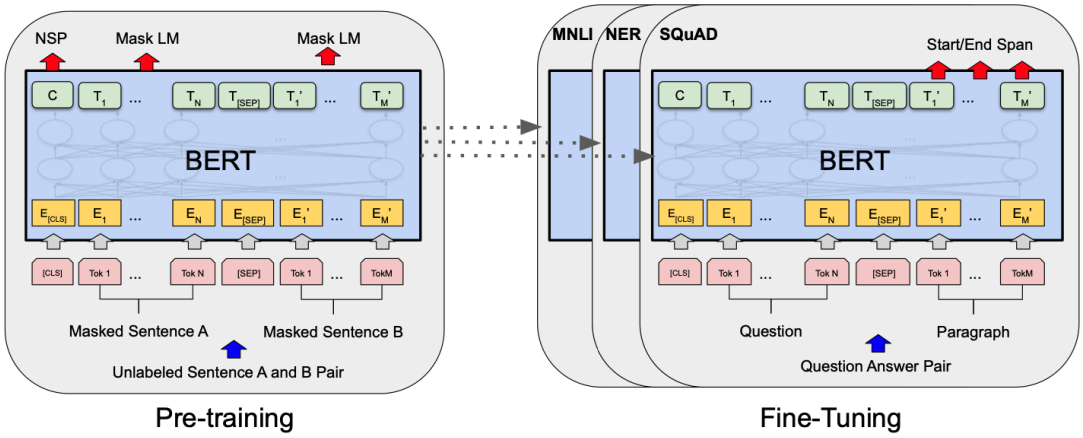


Figure 2.5: Pre-training and fine-tuning phases of BERT. Only the output layers are different in two architectures. Pre-trained model parameters can be used for any downstream task [23].

Task 2: Next Sentence Prediction. BERT architecture uses a second task for pre-training. This is motivated by the fact that some downstream tasks such as natural language inference or question answering are based on understanding the relationship between two sentences, which is not directly captured by language modeling. To make sure that the model learns sentence relationships as well, BERT is pre-trained for a binarized next sentence prediction (NSP) task. NSP is also an unsupervised task that can be trivially generated from any corpus without human annotations. For this task, 50% of the time, the second sentence actually follows the first (labeled as IsNext), and 50% of the time sentences are randomly selected from the corpus (labeled as NotNext).

Fine-tuning BERT. For fine-tuning BERT, we plug in the task-specific inputs and outputs and fine-tune all the parameters end-to-end. At the input, sentence A and sentence B from pre-training are analogous to (1) sentence pairs in paraphrasing, (2) hypothesis-premise pairs in entailment, (3) question-passage pairs in question answering, and (4) a degenerate text pair in text classification or sequence tagging. At the output, the token representations are fed into an output layer for token-level tasks, such as sequence tagging or question answering. A special token [CLS] is fed into an output layer for classification, such as entailment or sentiment analysis. Pre-training and fine-tuning phases of BERT are shown in Figure 2.5.

Chapter 3

Attributed Hierarchical Attention

3.1 Motivation

Patient-level classification is the task of predicting a clinical outcome of a patient (e.g.diagnosis, mortality risk, re-admission risk) based on the patient's Electronic Health Records (EHR). Clinical notes, a rich source of patient information found in EHR data, can be utilized for patient-level classification tasks as depicted in Figure 3.1.EHR notes are written by health-care professionals and often contain crucial details regarding patient care along with expert insights not found elsewhere in EHRs. Clinical notes have a nested sequential structure, namely they are a sequence of documents created over time and each document itself consists of a sequence of words. They are also accompanied by non-sequential meta-information at multiple layers. For example, the time at which a particular note is created or the category of the note (e.g.nursing, ECG) correspond to external attributes on the note level. Similarly, the demographic profile of the patient is patient-level information that does not change on the note level. Despite their value, usage of clinical notes in patient-level classification remains limited due to their complex multi-modal structure.

Standard text representation methods, such as bag-of-words or topic modeling, have been used to model clinical notes for patient-level classification [10, 31, 32, 44, 76, 89]. However, they are not able to capture the vast amount of information contained in notes, such as temporality of notes or semantic structure of words. More recently, aggregated vector representations of words have been explored to

represent notes and are then fed into linear models [27]. Vector representations of words leverage contextual information by encoding similar words into similar vectors [67, 73]. Despite incorporating contextual information, this method neglects the sequential nature at both the word and document levels (Figure 3.3-b). Bag-of-words representations of notes are fed into RNN models, capturing temporal dependencies between notes [27] . This approach makes use of the order of notes by using a sequential model (RNN) but does not consider the timing of documents or order of words (Figure 3.3-c).

A tremendous opportunity remains to design deep network models that capture the complex sequential dependencies at multiple layers, particularly within and among documents. An effective model for clinical notes must maintain long-term dependencies across multiple documents while at the same time accounting for a nested sequential structure and handling hierarchical external attributes. Recurrent Neural Networks (RNN) have recently shown great promise for document classification [62, 102, 114]. However, RNNs cannot be directly used on clinical notes. Instead, a custom-tailored architecture is required due to the following challenges:

- *Nested Sequential Structure*: Patient records contain time series of notes, each of which consists of a sequence of words. Hence they cannot be directly fed into an RNN without losing crucial sequential information inherent to at least one layer.

- *Long Term Dependencies*: Every note is typically a long document, composed of many hundreds of words. In practice, state-of-the-art text mining models still struggle to capture long-term dependencies. Even methods designed for this challenge are empirically difficult to optimize.

- *Multilayer Hybrid Data*: *Sequential* information is accompanied by external attributes at multiple layers. For example, every document is composed of not only a *sequence* of words but also a creation time, which is *static* across the sequence.

- *Required Interpretability*: Neural network-based text classification tends to be "black-box" in the sense that the models are challenging to interpret. However,

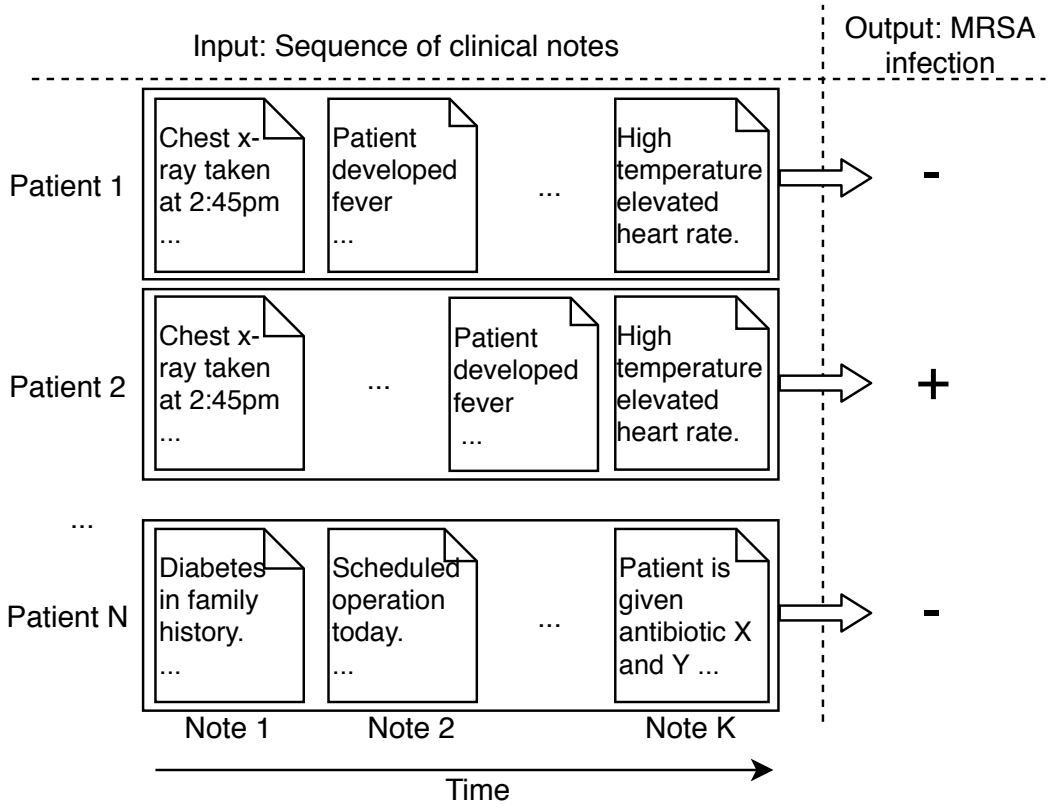


Figure 3.1: Patient-level diagnosis prediction using sequences of clinical notes. The task is to predict whether or not a patient will

in the clinical setting, interpretability is paramount to provide clinicians insights into how models choose particular predictions.

In this work, we design a hierarchical network to account for nested sequential document structures with external attributes at multiple levels for patient-level classification of clinical notes. In this proposed architecture, called HAC-RNN, sequences of words and series of documents are fed into different RNNs stacked hierarchically to account for the contextual information at the bottom layer and temporal information at the top layer without sacrificing representational power. Contextual and temporal attention mechanisms are incorporated to resolve three problems. First, they help to maintain long-term dependencies by selectively remembering early words in both long documents and documents created earlier in a patient's stay, while also achieving interpretability. Second, attention mechanisms control the information flow by deciding how much information to propagate

between layers. Finally, they integrate sequential and non-sequential data through conditioning on extrinsic attributes (e.g.the timing of a document at the note-level, a patient's age on the patient-level). Figure 3.3 illustrates a schematic of HAC-RNN compares it to state-of-the-art methods.

Our contributions are summarized as follows:

- We propose a hierarchical RNN with hierarchical attention conditioned on the external attributes for classification of *nested sequential document series*, with the first sequence being words and the second sequence being notes taken over time.
- With hierarchical attention, our model (1) learns nuanced levels of attention for individual words as well as for complete notes when constructing patient representations, (2) brings interpretability and provides insights by pointing to words, word phrases and notes that contribute to the classification decision, and (3) handles long sequences of both words and notes.
- With extensive experimental evaluation working on an in-hospital setting with real-world medical prediction tasks extracted from publicly available MIMIC dataset from Beth Israel ICU units, we validate that word order, document order, and external attributes improve the classification performance for patient-level classification. Our model achieves significant improvement over state-of-the-art models.

3.2 HAC-RNN Framework

Our proposed method, HAC-RNN, is composed of seven layers. An overview of HAC-RNN and each of its layers is presented in Figure 3.2. This section describes the details of the model.

3.2.1 Notation and Problem Definition

Clinical notes for each patient are represented as a sequence of text documents. The n-th patient of N total patients is represented by a sequence of $L^{(n)}$ documents and a vector $\gamma^{(n)}$ that contains patient-level attributes, such as age. γ is not

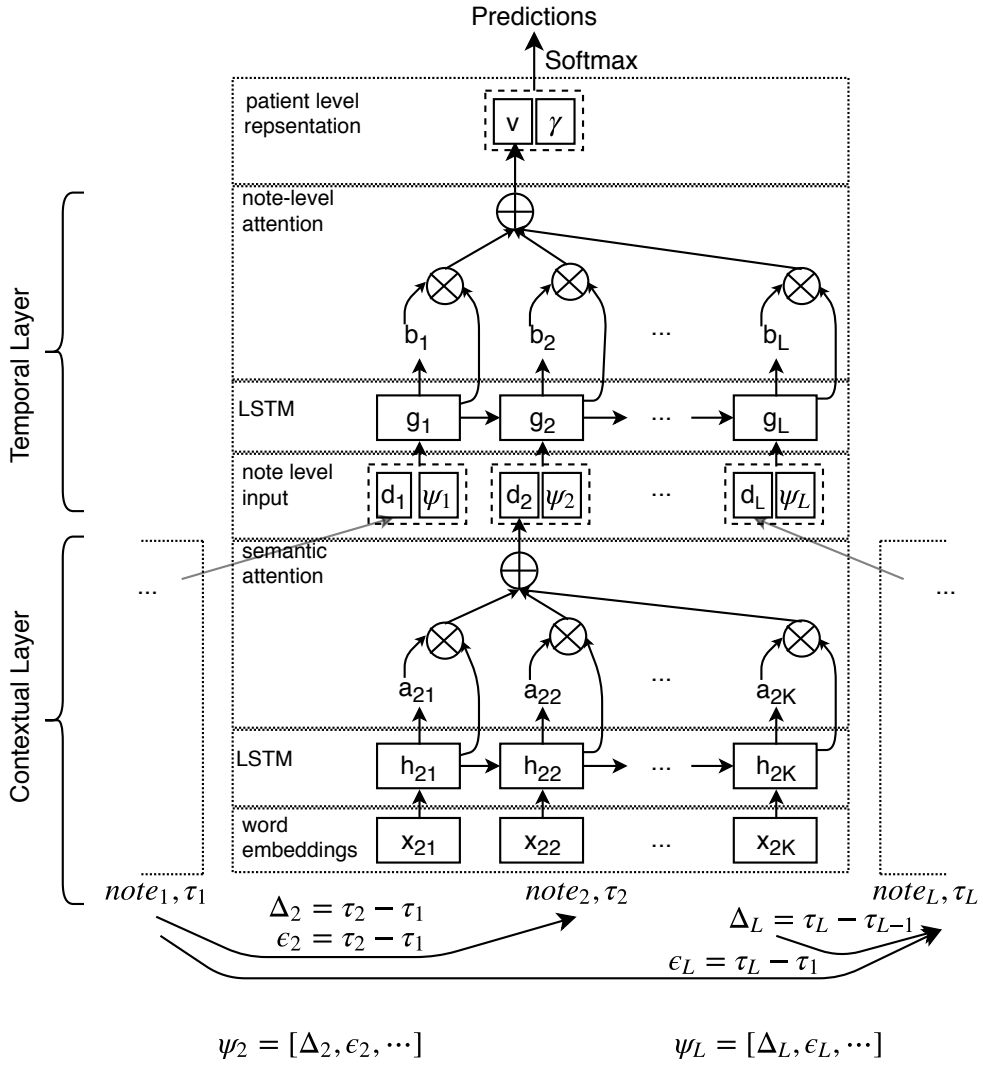
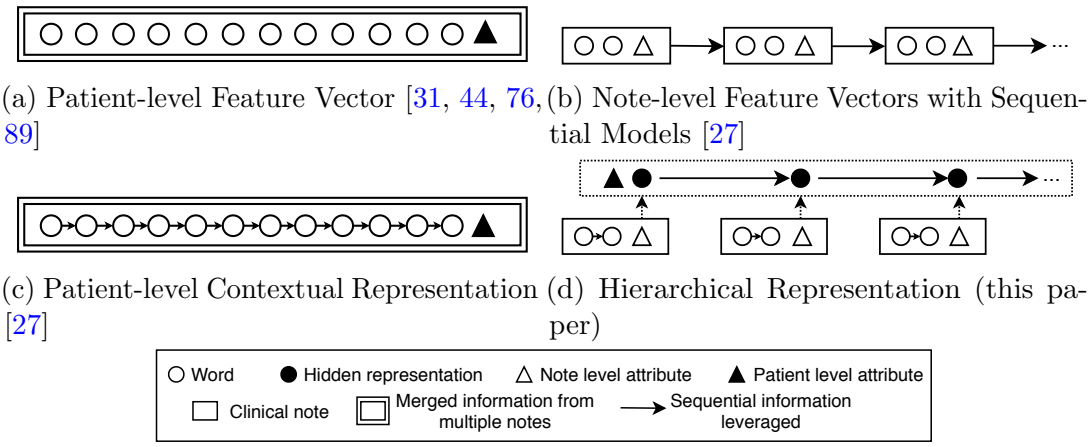


Figure 3.2: Architectural design of HAC-RNN.



(a) Patient-level Feature Vector [31, 44, 76, 89]

(b) Note-level Feature Vectors with Sequential Models [27]

(c) Patient-level Contextual Representation [27]

(d) Hierarchical Representation (this paper)

Figure 3.3: Comparison of state-of-the-art methods to our model. (a) LDA or BOW are used to create a single feature vector for the patient. No encoding of sequential information on word or note level. (b) LDA or BOW are used to create a feature vector per note, with a sequential model. Captures the sequential ordering of notes. (c) Vector representations of words, aggregated into a patient level feature vector. Encodes the contextual information of words, neglects the order of notes. (d) HAC-RNN utilizes the sequential information on both word and note levels, as well as hierarchical external attributes.

extracted from a clinical note, but instead it is external meta information within the data source. A note is a tuple $(\mathbf{d}_i^{(n)}, \psi_i^{(n)})$ where $i = 1, \cdots, L^{(n)}$, with $\mathbf{d}_i^{(n)}$ denoting the i-th document of the sequence and $\psi_i^{(n)}$ denoting the meta-vector. Each document $\mathbf{d}_i^{(n)} \in \mathbb{R}^{K_i^{(n)}}$ contains $K_i^{(n)}$ words $w_{ij}^{(n)}$, $j = 1, \ldots, K_i^{(n)}$, $i = 1, \cdots, L^{(n)}$, and the meta vector $\psi_i^{(n)}$ carries external note-level attributes, such as creation time or category of the note.

The goal is to learn a model that predicts the label $\hat{y}^{(z)} \in \{0, 1\}$ for a new patient, given the set $\mathcal{D} = \{\mathcal{D}^{(1)}, \cdots, \mathcal{D}^{(n)}\}$, where $\mathcal{D}^{(n)} = \{\langle(\mathbf{d}_1^{(n)}, \psi_1^{(n)})), \ldots, (\mathbf{d}_{L^{(n)}}^{(n)}, \psi_{L^{(n)}}^{(n)})\rangle, \gamma^{(n)}\}$ and true labels $y^{(n)} \in \{0, 1\}$ (e.g.1 indicates an infection, 0 not-infection). For simplicity and readability, we henceforth describe our method for a single patient and thus drop the superscript (n) whenever it is unambiguous.

3.2.2 Embedding Layer

The input to HAC-RNN is a sequence of real-valued vector representations of words where semantically similar words are mapped close to one another. We learn

word embeddings prior to end-to-end model training disjointly, using the skip-gram model of the "word2vec" algorithm [67] as described in Section 2.3. Using the learned embedding vector θ, words are mapped to vectors $\mathbf{x}_{ij} = \theta w_{ij}$.

3.2.3 Contextual Layer

3.2.3.1 Sequential Input

We use a hierarchical approach to create patient level representations of document sequences. The bottom level of this hierarchy, referred to as the contextual layer, is designed to encode the contextual structure of the words within each individual document.

Given embedding vector $\mathbf{x}_{ij}$ of the word w_{ij}, where $i = 1, \cdots, L$ and $j = 1, \cdots, K_i$, we use an LSTM to summarize the information within each document. The LSTM reads the input word vectors from $\mathbf{x}_{i,1}$ to $\mathbf{x}_{i,K_i}$ from the document i and calculates a sequence of hidden states.

$$h_{ij} = \text{LSTM}(\mathbf{x}_{ij}), j \in [1, K_i] \tag{3.1}$$

Afterwards, a semantic attention at the word level is applied. The goal of this word-level attention mechanism is to identify which words contribute most to the meaning of the document. We first use a fully connected layer to get u_i as a hidden representation of h_{ij}.

$$u_{ij} = \tanh(W_w h_{ij} + b_w) \tag{3.2}$$

Then we define a context vector u_w, which can be thought of a high level representation or summary of the document. u_w is randomly initialized and jointly learned. The importance of each word is measured by the similarity of u_{ij} with the word level context vector u_w. We compute normalized importance weights α_{ij} through a softmax function as shown in Equation 3.3:

$$\alpha_{ij} = \frac{\exp(u_{ij}^\top u_w)}{\sum_j \exp(u_{ij}^\top u_w)} \tag{3.3}$$

Finally, the document vector representation is the weighted sum of word annotations and importances:

$$\mathbf{d}_i = \sum_j \alpha_{ij} h_{ij} \tag{3.4}$$

3.2.3.2 External Attributes

Besides the content of the note, a set of meta-information about the note itself is available. First, from the raw timestamp information τ_i that represents the time that the note was created, we compute a set of time variables. These variables include *delta time* Δ_i, hours between two consecutive notes, and *ordinal time* ϵ_i, hours passed since the patient's admission to the hospital. Thus multiple time semantics are represented. The next meta-information is the category of the note, (e.g.nursing report or ECG). This is encoded as a one-hot vector. We combine these meta-information sources into one note-level attribute vector ψ_i. Typically, external attributes are used for conditioning the LSTM via one of the two approaches [38]. The first is to use conditioning data as input to the hidden state by either addition $h_{ij} = \text{LSTM}(x_{ij} + \psi_i, h_{ij-1})$ or stacking $h_{ij} = \text{LSTM}\left(\begin{bmatrix} x_{ij} \\ \psi_i \end{bmatrix}, h_{ij-1} \right)$. The second option is to connect the attributes to the output $[h_{ij}, \psi_i]^\top$. Note that addition requires the vectors to have the same dimensionality. The stacking method can be quite costly, given that it increases the size of several matrices. When an attention mechanism is applied on top of the LSTM layer, output method becomes as costly since the attention mechanism consumes LSTM outputs from all timesteps. For computational optimization, we use the output method, replacing LSTM output with attention output $[\mathbf{d}_i, \psi_i]^\top$. This can be thought of conditioning the attention instead of LSTM with the note-level attributes. This combination is fed to the next layer of hierarchy.

3.2.4 Temporal Layer

3.2.4.1 Sequential Input

After computing the document level representations, another recurrent network learns to represent this sequence of representations. It reads the input document vectors, each being a dense encoding of content, time, and category, then calculates a sequence of hidden states.

$$h_i = \text{LSTM}(\mathbf{d}_i), i \in [1, L] \tag{3.5}$$

We observe that not all the documents are equally relevant to the prediction

task. Hence we apply a temporal attention to selectively attend to note instances that are more relevant as follows:

$$u_i = \tanh(W_d h_i + b_d) \tag{3.6}$$

$$\alpha_i = \frac{\exp(u_i^\top u_d)}{\sum_t \exp(u_i^\top u_d)} \tag{3.7}$$

Finally, the patient level representation is the weighted average of the hidden vectors of documents with attention weights.

$$\mathbf{v} = \sum_i \alpha_i h_i \tag{3.8}$$

3.2.4.2 External Attributes

Similar to the previous layer, EHR data has patient-level attributes that exhibit a static structure across each patient's stay. In our model, we use age of the patient, γ, as conditioning data. Referring to the same logic with the contextual layer, we use external attributes as $[\mathbf{v}, \gamma]^\top$.

3.2.5 Patient Classification

The resulting vector $[\mathbf{v}, \gamma]^\top$ can be seen as a dense embedding for the patient. The final layer takes this as input and assigns a probability to each possible class. We use the cross-entropy loss function as the objective function where $\hat{y}$ is the prediction and y is the ground truth label, shown in Equation 6.7.

$$L = -(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})) \tag{3.9}$$

3.3 Experimental Evaluation

3.3.1 Medical Notes Dataset & Prediction Tasks

For our experimental evaluation, we use the publicly-available critical care database MIMIC III [46]. MIMIC III was collected from the Beth Israel Deaconess Medical Center Intensive Care Unit (ICU) between 2001 and 2012. It contains all unstructured notes taken by the caregivers of 45,000 patients. MIMIC is a benchmark dataset frequently used in clinical machine learning studies [9, 22, 37,

84, 90]. We experiment with three prediction problems using relevant cohorts extracted from the MIMIC dataset.

- **Clostridium Difficile Infection Prediction**: Clostridium difficile infection (CDIF) is a common hospital acquired infection (HAI) resulting in gastrointestinal illness with high morbidity and mortality rates. To identify CDIF patients in the MIMIC, we use the microbiology test associated with the organism *80139-Clostridium Difficile* found in the *Microbiology Events* table. Of 1,079 CDIF-positive patients, 1,035 have note events. These patients form the CDIF-positive population. We randomly subsample 1,035 patients with no record of this test, obtaining a balanced dataset.

- **MRSA Infection Prediction**: Methicillin-resistant Staphylococcus aureus (MRSA) is a common cause of serious HAIs. We use the microbiology test associated with the organism *80293-MRSA*. Of 1,304 MRSA-positive patients, 1,240 have note events. These patients formed our MRSA-positive population. We then randomly subsample 1,240 patients who have no record of a test for organism 80293, obtaining equally-sized groups of positive and negative examples.

- **In-hospital Mortality Prediction**: Detecting high-death-risk patients is a benchmark task in clinical predictive modeling research [45, 75]. We use *hospital_expire_flag* from the *Admissions* table to extract mortality-positive patient cohort. 5,854 patients have *hospital expire flag* $= 1$, indicating that they died in the hospital. 5,000 of these patients with note events prior to their death are used as our mortality-positive cohort. We then randomly subsample 5,000 patients who have *hospital expire flag* $= 0$, forming the mortality-negative cohort.

We detail statistics for each dataset in Table 3.1. For all three datasets, we only use data up to a day before diagnosis (CDIF & MRSA) or death to assure no information regarding confirmation or treatment of the condition is included. For negative patient sets, half-way of their data is used as usually done in prior work [90, 109].

Many previous works on clinical note modeling embrace a multiclass classification setting where they predict ICD10 diagnosis codes [68, 77, 94]. Even though

automatic extraction of diagnosis codes are very important, these systems cannot be used to assist clinical decision making in in-hospital settings. For inpatients, ICD10 codes are assigned to patients at the end of their hospital stay, thus, these models are inherently trained on the data that contains information not only before the confirmation of a diagnosis, but also after it (i.e.related to treatment). In this paper, we adopt an experimental setting that allows us to predict a medical condition *before* it happens which enables real-world applications.

3.3.2 Implementation Details

3.3.2.1 Data Preparation

There are notes from 15 categories in MIMIC. These categories are Case Management, Consult, ECG, Echo, Discharge summary, General Nursing, Nursing/ other, Nutrition, Pharmacy, Physician, Radiology, Rehab Services, Respiratory, Social Work. We use notes from all categories except for discharge summaries, as the later are only created at the end of the hospital stay. We only will use EHR data before the diagnosis of the condition. We also note that the current literature is heavily based on discharge summaries which are semi-structured, unlike the other categories.

We transform the clinical notes into lowercase and remove punctuations. As the MIMIC dataset contains some de-identified data; sensitive information, such as a doctor or patient name, is deducted and put in square brackets. (e.g., Attending:[**First Name3 (LF) 1**]). We remove these parts from the notes. The creation-time of each note is used to extract time features. Lastly, attributes are extracted from other tables, such as the *age* attribute from the *Patients* table.

3.3.2.2 Model Training

For all 3 cohorts, we learn local word embeddings using all notes that belong to patients from the training set. We use the skip-gram architecture of Word2vec [67] to learn vector representation of words. We set the skip window size to 1, number of skips to 2, and number of negative examples to sample to 64. Our embedding size is 32. Based on the mean/median number of notes and words (see Table 3.1), we use 20 notes per patient and 300 words from each note. We use the Adam

Table 3.1: Dataset Statistics

Statistics		Dataset		
		CDIF	MRSA	Mortality
# Notes / Patient	Mean	32	14	30
	Median	14	7	19
# Words / Note	Mean	339	379	273
	Median	207	218	189
Total # Notes		66,486	35,332	299,256
Total # Patients		2,070	2,480	10,000

optimizer [50]. Exponential decay learning rate schedule is used with an initial rate of 0.01. Model performance is measured by accuracy. We implement our model with Tensorflow [2].

3.3.3 Experimental Design

We conduct a four-pronged experimental evaluation to asses the prediction power stemming from (1) the sequential information on the word-level, (2) the sequential information on the note-level, (3) incorporating nested sequential information from both levels, and (4) multilayer attribute conditioning. We fix the number of words per note and the number of notes per patient across all experiments to ensure inputting the same amount of information to all models allowing a fair comparison. Additionally, we run a set of experiments to compare HAC-RNN's performance with state-of-the-art models.

3.3.3.1 Word-level Models

We first build and analyze a set of word-level models that capture the word order but neglect the sequential order of notes. To this end, for each patient, we concatenate the final 300 words from each note, forming one large document per patient. We use *final* words as opposed to *first* because the beginning of a note often includes repeated information from previous notes. In the case that a note

Table 3.2: Performance comparison of baseline methods and the proposed method. (Test-set Accuracy)

Method Type	Compared Methods	Datasets		
		CDIF	MRSA	Mortality
Baselines	Bag of Words + SVM	58.50 ± 1.37	68.64 ± 0.91	71.07 ± 1.79
	Bag of n-grams + SVM	59.72 ± 2.46	69.20 ± 0.28	68.50 ± 2.46
	LDA + SVM [89]	58.54 ± 1.66	69.56 ± 0.95	71.08 ± 0.69
	Average embeddings [27]	58.79 ± 0.45	70.50 ± 0.80	74.76 ± 0.63
	Sequential BOW + RNN [27]	61.56 ± 1.78	71.08 ± 1.12	77.52 ± 1.18
Word-level models with word embeddings	Fully Connected	51.47 ± 0.66	49.32 ± 0.61	50.60 ± 0.75
	RNN	55.61 ± 0.33	58.26 ± 1.85	60.69 ± 1.14
	RNN Attention	64.90 ± 1.19	72.58 ± 0.40	76.52 ± 0.66
Note-level models models with average embeddings	Fully Connected	58.79 ± 0.45	70.50 ± 0.80	74.76 ± 0.63
	RNN	58.78 ± 1.19	68.2 ± 1.70	71.85 ± 1.69
	RNN Attention	61.17 ± 0.10	71.62 ± 1.43	73.63 ± 1.28
	RNN Attention + time	62.30 ± 0.71	75.52 ± 1.19	76.24 ± 0.33
Hierarchical models with word embeddings	H-RNN	57.24 ± 1.72	71.02 ± 0.59	63.41 ± 1.80
	HW-RNN (Word Attention)	66.46 ± 1.47	74.20 ± 0.38	82.99 ± 1.09
	HA-RNN (Double Attention)	$\mathbf{67.68} \pm 1.64$	74.77 ± 0.71	84.25 ± 0.61
	HAC-RNN (Multilayer Conditioning)	65.61 ± 0.93	$\mathbf{77.57} \pm 1.43$	$\mathbf{85.33} \pm 0.75$

does not have 300 words, we pad the beginning of the note with a vector of 0's to ensure same-length inputs. We then train both a Fully Connected Network and an LSTM. We hypothesize that the sequential model (LSTM) will work better as it captures the semantic structure of words and word phrases.

3.3.3.2 Note-Level Models

We also build a set of note-level models where we create a set of note-level representations, then use these representations to classify patients. For each patient, we use the 20 notes leading up to the time-of-prediction (e.g., the laboratory test confirming MRSA). In the case that a patient has fewer than 20 documents, we pad their sequences up to 20 with vectors of 0's. If a patient has more notes, first notes are discarded to keep only 20. This decision is based on the intuition that more recent notes should carry more relevant information. For example, CDIF and MRSA are caused by bacteria with short incubation periods [37, 90] which means symptoms should appear nearing diagnosis. We calculate the coordinate-wise mean vector of all words within each note, producing 20 fixed-length vectors, one per note. These resulting vectors are then fed into a sequential model. This model takes into account the order of notes while ignoring the order of words. We also

compute the average vector of all notes, constituting one fixed-length vector per patient. Unlike the first model, this model ignores the temporal information of note order.

3.3.3.3 Hierarchical Models

Finally, hierarchical models take into consideration the nested-sequential order of words and notes. Our hierarchical architecture employs two levels of RNN to account for contextual and temporal information, called H-RNN. In addition to H-RNN, we incorporate an attention mechanism, first only at the word-level (HW-RNN), then at both levels (HA-RNN). A fourth model is learned using multilayer conditioning on the external attributes (HAC-RNN).

3.3.3.4 Compared Methods

We also implement current state-of-the-art methods in clinical text mining.

- *Bag-of-words representations*: We combine patient notes into one document. Using the most-informative 2000 words, a bag-of-words representation is created. A linear SVM is then trained. Bag-of-words representation with linear SVM is utilized in many works [10, 44, 76].

- *Bag-of-n-grams representations*: We use bag of unigrams, bi-grams, and trigrams with the most informative 2000 phrases. A linear SVM is then trained.

- *LDA topic model*: Using Latent Dirichlet Allocations, we represent patient notes as weighted averages of a set of topics. We keep the number of topics to be 50 and train a linear kernel SVM. This method is commonly used to model clinical notes [31, 32, 43, 100].

- *Embedding Averaging*: Learned word embeddings of each word are averaged, then the average of these note-level averages is computed and fed into a linear classifier [27].

- *Sequential bag-of-words*: Each note is represented as a bag-of-words, resulting in a sequence of note-level representations. These representations are then fed into an RNN for final prediction [27].

3.3.4 Results

We present test set accuracy of all experiments in Table 3.2.

Word-Level Models For all three tasks, sequential models outperform non-sequential models demonstrating contextual information is vital for classification. Moreover, RNN with attention outperforms RNN without attention for all tasks by large margins (0.09, 0.14, 0.15 for CDIF, MRSA, Mortality respectively). This confirms that attention mechanism is extremely beneficial when dealing with long input sequences.

Note-Level Models For CDIF and MRSA tasks, sequential models (with attention) with note-level input achieved better results compared to patient-level representation, showing document order is informative for the classification task. However, for Mortality prediction, non-sequential model achieved a slightly better result. Another important observation is that for all datasets, RNN with attention performs better compared to without attention. Even though the difference is small, RNN models get worse results compared to fully connected models. In these two experiment sets, embedding vectors are coordinate-wise averaged, which may be the reason for this result. As for conditioning data, only note-level time information is used here to observe the effect of the timing of the notes. These experiments outperform all the others among note-level models, proving the importance of temporal information.

Hierarchical Models While HAC-RNN achieved the best performance for MRSA and Mortality tasks, HA-RNN outperforms for CDIF. This proves that both word and note order is paramount to classification task whereas patient profiles may not be as vital depending on the condition. All hierarchical attention models (HW-RNN, HA-RNN, HAC-RNN) outperform all other baselines for CDIF and Mortality, whereas for MRSA, note-level RNN Attention with time is the second best performing model. This shows that, depending on the disease characteristics, temporal information or the exact time of the notes may be more relevant to the classification task.

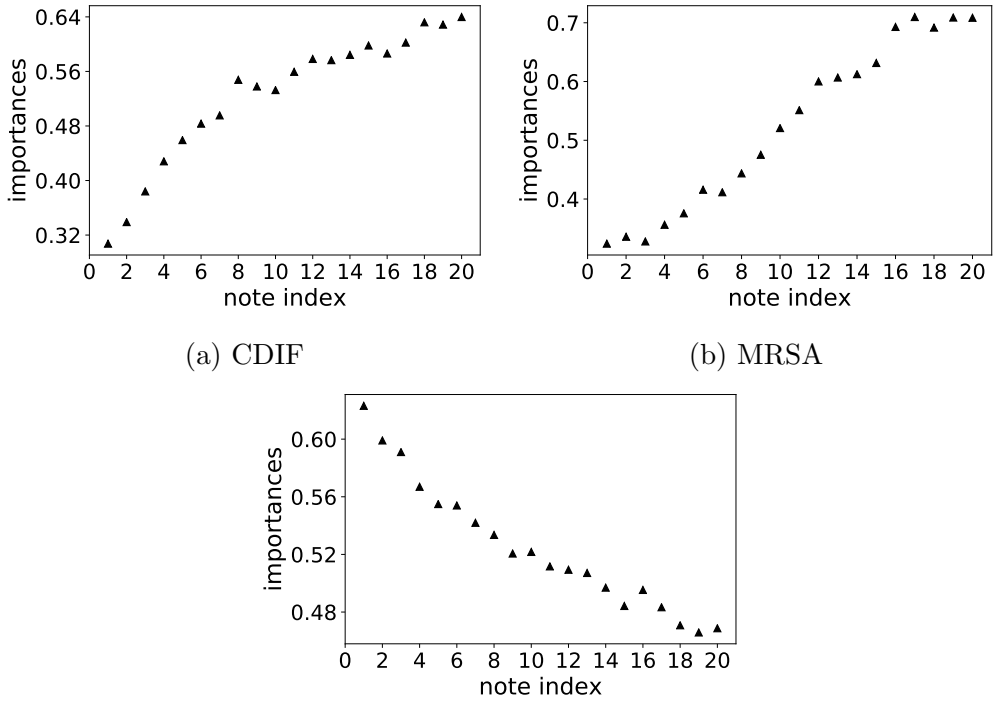
3.3.5 Model Interpretation

HAC-RNN not only accomplishes superior classification performance but provides an interpretable model. One essential problem with deep learning in the clinical domain is the black-box nature of deep learning algorithms. Since these methods are not able to provide explanations or intuition for particular classification decisions, health-care professionals approach usage of these algorithms warily in the decision-making process. Thus, interpretable deep learning models are imperative. We examine the attention weights generated by HA-RNN in the following subsections.

Note-level attention We first present results from the attention layer at the note-level. The attention weight associated with a note indicates how much the algorithm emphasizes it when generating labels. This weight is always between 0 and 1 and the weights across all notes of a patient sum to 1. We rank the attention weights for the notes of each testing patient from most-attention to least-attention, 20 corresponding to the highest rank (i.e.the most important note). To compute an importance score, we then average the rankings per note across the whole patient set and scale values into a 0-1 range. Results are presented in Figure 3.4.

We observe that for the MRSA and CDIF datasets, more recent notes receive more attention. Both MRSA and CDIF are hospital-acquired infections with short incubation periods (3-7 days), a fact with which our results are consistent. For Mortality Prediction, there is an opposite trend in data, where initial notes receive more attention. This information can be used to direct the clinicians to guide them on which notes to focus on. For example, for CDIF we recommend reading the last couple of notes while for Mortality prediction we recommend reading the earlier notes, (i.e.those closer in time to the admission of the patient).

Word-level attention Figure 3.5 displays the most important words from the most important notes of a successfully-classified patient from Mortality prediction cohort who died while in the hospital. The yellow shading indicates the amount of attention a word receives where darker yellow means more attention. These word importances can assist clinicians with where to look for additional clues in the notes. As Figure 3.5 shows, our algorithm assigns high importance to state changes ('gtt increased') and medical conditions ('resp distress hypotension sepsis', 'femoral introducer', 'dialysis'). Our algorithm also learns to pay attention to *expert insights*



(a) CDIF

(b) MRSA

(c) Mortality

Figure 3.4: Importances of notes averaged over the patients. For CDIF and MRSA, the most recent notes tend to be more relevant to the classification result, while for Mortality prediction earlier notes tend to be more important.

such as 'watch qtc level'.

3.4 Summary

Clinical notes present a nested sequential structure, namely, for each patient, there is a series of free-form text documents (notes) over time and each document itself consists of a sequence of words. These notes are accompanied by external attributes at each level of granularity. In this paper, we propose an Attributed Hierarchical Attention network with multiple attention mechanisms conditioned on external attributes at different layers for predictive modeling of the sequence of clinical notes. We evaluate our method on three distinct clinical prediction tasks, namely, Clostridium Difficile Infection prediction, MRSA infection prediction, and in-hospital mortality prediction. Patient cohorts are extracted from the publicly-

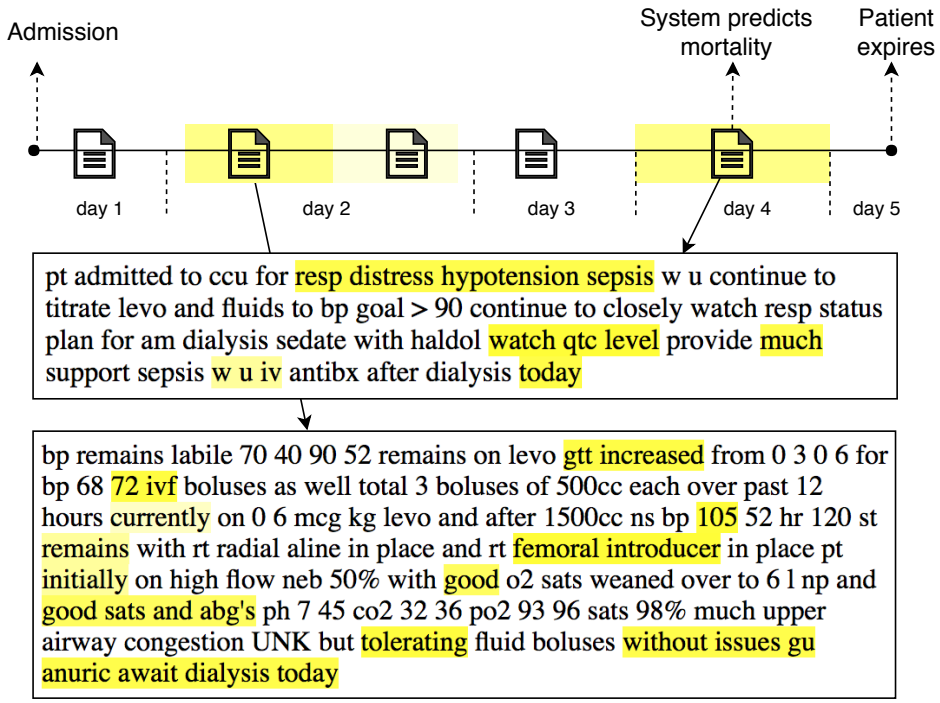


Figure 3.5: Example notes for a correctly-predicted test patient with most important words/notes highlighted. Background shades within the text represent word attention. Darker shades correspond to higher attention weights (This figure best viewed in color).

available Electronic Health Records data from Beth Israel Medical Center (MIMIC-III database). We extensively evaluate our method's prediction performance on these three tasks. We conclude that considering word-order, note-order, and the combination of the two outperform current state-of-the-art methods for clinical note classification. Moreover, external attributes are also shown to be beneficial by utilizing inferred patient profiles for achieving improved prediction. By including attention mechanisms, our method has been shown to recommend either whole-notes or specific sentences for clinicians to spend their valuable time reading.

This task is resulted in the following publication:

- C. Sen, T. Hartvigsen, X. Kong, E. Rundensteiner, "Patient-level Classification on Clinical Note Sequences Guided by Attributed Hierarchical Attention", In 2019 IEEE International Conference on Big Data (Big Data), pp. 930-939. IEEE, 2019.

Chapter 4

Time-Aware Dual Attention

4.1 Motivation

Clinical notes, collected as part of Electronic Health Records (EHR), are timestamped sequences of documents created by health-care professionals during a patient's hospital stay or over the course of several visits. The aim of clinical note classification is to predict a clinical outcome for a patient, such as a future diagnosis, in-hospital mortality, etc., using the clinical notes as input to a classification model. While they contain rich content full of expert insights about patients, these free-form textual documents are more challenging to mine than structured data types. Furthermore, they correspond to a *timed* sequence of documents. Here, we observe that the time at which a clinical note was created can carry critical importance. For example, consider two in-patients with clinical notes containing identical content as illustrated in Figure 4.1. While the first patient is diagnosed with a life-threatening infection, the second patient is discharged without contracting this infection. A machine learning model taking into account only the content of the notes would result in a wrong prediction. In this example, the timing of the clinical notes provides the discriminative power vital for accurate classification.

Numerous designs incorporating time information into sequential text modeling networks such as Recurrent Neural Networks (RNN) are possible. For instance, a modified RNN is proposed to account for time, discounting short term memory proportional to the amount of time passed from the previous timestep for patient subtyping task [9]. Since RNN models lack interpretability, pairing them with

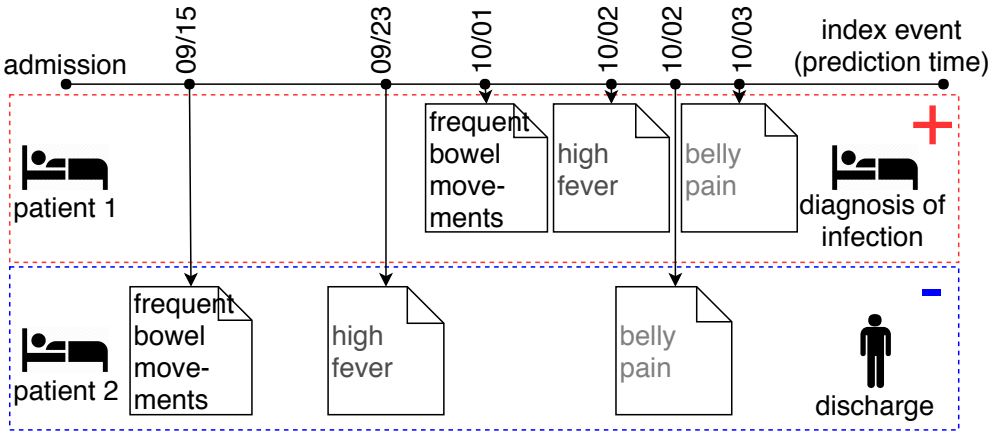


Figure 4.1: Clinical note sequences from two patients with identical content but distinct time of occurrences. While first patient (top) contracts an infection, the other (bottom) does not. Here, classification based only on the content of notes will be wrong; while timing of the notes adds information critical for classification.

attention mechanisms has begun to be studied in the medical domain [15, 16, 65]. Recently, using time in the form of an attention mechanism that allows the model to pay more or less attention to the input based on associated timestamps is proposed for spoken language understanding [14, 96]. In [14], the hard-coded function $1/d(t)$ is hand-picked, with $d(\cdot)$ being the time difference between two instances, as the attention weight at each time step. In [96], a parameterized time-decay function $d(t)$ generates the attention weights. While the latter is more flexible compared to the former, it still makes a strict assumption about the functional form of the time attention.

These methods assume that as time passes, old information becomes less relevant. However, this assumption does not always hold, especially in the clinical domain. For example, consider clinical notes collected over the hospital-stay for an inpatient. A note created at the admission time contains patient information such as age and gender and thus remains relevant to an infection prediction task regardless of how much time passes. On the other hand, the importance of medical events (e.g.having a surgery, elevation in heart rate) relative to a prediction task depends more heavily on the exact occurrence time of events. Furthermore, the relevance of each medical event to a medical outcome (e.g.infection) changes differently as time passes. Figure 4.2 presents examples of how temporal importance may change over

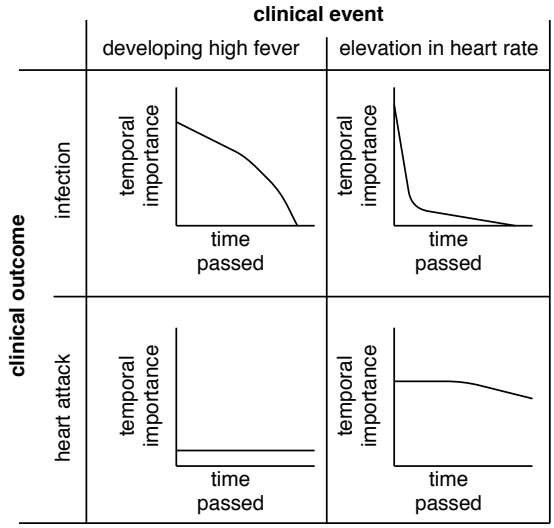


Figure 4.2: Temporal importance is a function of both clinical event and clinical outcome.

time for different event-outcome pairs. Overall, a "temporal importance function" for EHR note classification depends on two factors: 1) characteristics of the medical outcome that is being predicted, and 2) medical events happening to the patient throughout their hospital stay (i.e.content of the note).

Designing a time-aware attention mechanism that holds the aforementioned properties remains challenging for the following reasons :

- *Unknown temporal importance*: The true function relating *when notes are recorded* and *clinical outcomes* is naturally unsupervised. Additionally, this relationship may change per task, so defining one fixed function is too rigid. Previous works input a single time representation to their model (time difference between two consecutive time steps); while other time representations may much better match the underlying relationship between the time occurrence of instances and how much attention they should receive.

- *Required Interpretability*: "Black-box" models are suspicious to many clinicians. Previously proposed hard-coded temporal importance functions or RNN modifications do not support interpretability since they are pre-set instead of being learned alongside a classification model.

- *Balancing multiple sources of attention*: The goal is not only to decide how

much attention to put on each document based on their content but also based on the time at which they were created. Instead of rigid combinations of the two, a balance among them must be *learned* as the correct balance is task-dependent.

To address these challenges, we propose a novel attention mechanism, henceforth called TEND for Time ENhanced Dual attention, that leverages both the content and the timing of clinical notes. We pair TEND with an LSTM, a powerful sequence-modeling neural network, resulting in TEND-LSTM, which classifies sequences of clinical notes. To account for both factors of temporal importance, TEND-LSTM computes a multilayer attention mechanism. On the first layer, it focuses on the content of a single note which is represented as an abstract summarization of clinical events. Then on the second layer, using a number of time representations, it learns a task-dependant time attention integrated with content attention. An MLP layer is used to combine content and time attention. Our contributions can be summarized as follows:

- We propose a flexible end-to-end model for patient outcome prediction using clinical note sequences called TEND-LSTM. To our best knowledge, this is the first work proposing a time-informed attention mechanism for sequential document classification.
- Our attention mechanism, TEND, does not make any assumptions about the functional form of the time attention, instead inferring it from examples.
- TEND automatically learns to balance the content and timing of notes when learning attention weights, offering additional flexibility.
- TEND brings a broader level of interpretability by offering explanations about the decision of the model from both the time and content perspectives.
- We demonstrate TEND-LSTM outperforming existing methods on six real-world medical prediction tasks using real-world EHR datasets by large margins.

4.2 TEND-LSTM Framework

In this work, we design a novel time-augmented dual attention mechanism addressing the problems with the existing work for classifying document sequences. The overall architecture of TEND-LSTM is presented in Figure 4.3. This section introduces the details of each layers of TEND-LSTM.

4.2.1 Notation and Problem Definition

Clinical notes for each patient are represented as a sequence of text documents collected up to an index event. In our use case, the index event may be the diagnosis of the infection for the patients. The n-th patient of N total patients is represented by a sequence of $L^{(n)}$ time-document tuples $(\mathbf{d}_i^{(n)}, t_i^{(n)})$, with $\mathbf{d}_i^{(n)}$ denoting the i-th document of the sequence and $t_i^{(n)}$ denoting the timestamp at which this document was created where $i = 1, \cdots, L^{(n)}$. The document $\mathbf{d}_i^{(n)} \in \mathbb{R}^{K_i^{(n)}}$ contains $K_i^{(n)}$ words $w_{ij}^{(n)}$, $j = 1, \ldots, K_i^{(n)}$.

From the raw timestamp information $t_i^{(n)}$, we compute a set of time variables and store them in a time vector $\psi_i^{(n)}$. Thus we represent time information with respect to different time semantics such as admission time or the time of the index event and in different granularities such as hours or days. The goal is to learn a model to predict the label for a new patient $\hat{y}^{(z)} \in \{0, 1\}$, given the set of $\mathcal{D} = \{\mathcal{D}^{(1)}, \cdots, \mathcal{D}^{(n)}\}$, $\mathcal{D}^{(n)} = \langle(\mathbf{d}_1^{(n)}, \psi_1^{(n)})), \ldots, (\mathbf{d}_{L^{(n)}}^{(n)}, \psi_{L^{(n)}}^{(n)})\rangle$ and true labels $y^{(n)} \in \{0, 1\}$ (e.g., 1 indicates an infection, 0 not-infection). To simplify readability, we henceforth describe our method for a single patient and thus drop the superscript (n) whenever it is unambiguous.

4.2.2 Document Representations

We compute a sequence of real-valued vector representations for each patient, one vector per document as follows.

Mean word embedding for content summarization Word embeddings are continuous vector representations of words where semantically similar words are mapped close to one another in a new low-dimensional space. We learn word embeddings prior to end-to-end model training disjointly, using the skip-gram model

Table 4.1: Basic Notation

Notation	Explanation
N	Total number of patients
$L^{(n)}$	Total number of document-time tuples for patient n
$\mathbf{d}_i^{(n)}$	i-th document of patient n
$t_i^{(n)}$	Timestamp associated with the i-th document of patient n
$\psi_i^{(n)}$	Computed time vector associated with the i-th document of patient n
$K_i^{(n)}$	Total number of words in the i-th document of patient n.
$w_{ij}^{(n)}$	j-th word in the i-th document of patient n
$\mathbf{x}_{ij}^{(n)}$	Vector representation of word $w_{ij}^{(n)}$
$\mathbf{e}_i^{(n)}$	Document level representation of $\mathbf{d}_i^{(n)}$
$\tau_i^{(n)}$	Time-enhanced document vector
	where $i = 1, \ldots, L^{(n)}$ and $j = 1, \ldots, K_i^{(n)}$.

of the "word2vec" algorithm [67] as described in Section 2.3. Using the learned embedding vector θ, words are then mapped to vectors using $\mathbf{x}_{ij} = \theta w_{ij}$, $\mathbf{x}_{ij} \in \mathbb{R}^{\omega}$. Embedding size ω is a parameter to be optimized during training.

After computing the $\mathbf{x}_{ij}$ mapping for each word $\mathbf{w}_{ij}$, a document is now a sequence of vectors $\mathbf{d}_i = [\mathbf{x}_{i1}, \ldots, \mathbf{x}_{iK_i^{(n)}}], \mathbf{x}_{ij} \in \mathbb{R}^{\omega}$. We take the coordinate-wise average and compute a single vector for each document $\mathbf{e}_i \in \mathbb{R}^{\omega}$ as follows:

$$\mathbf{e}_i = \left[\left(\frac{1}{K_i^{(n)}} \sum_{j \in K_i^{(n)}} \mathbf{x}_{ij}[1] \right), \cdots, \left(\frac{1}{K_i^{(n)}} \sum_{j \in K_i^{(n)}} \mathbf{x}_{ij}[\omega] \right) \right] \tag{4.1}$$

Time enhanced document vector When generating a time representation, appropriate granularity depends on the data source. While a medical sensor may produce data every second to minute, clinical notes are recorded much less frequently, in the range of once an hour to a day (or a few days). Consider the case where a patient has two notes taken on the second and the fourth hours of their stay. Representing time as $(2, 4)$ (second and fourth hours) or $(1, 1)$ (both are taken on the first day) may be both meaningful from a medical perspective depending on the note content while leading to different prediction results.

Time representation can be absolute or relative to a reference point. For example, an admission time note, containing a patient's profile, might always be important regardless how much time has passed. On the other hand, having a high fever is a symptom of a condition only for a short period of time. If that condition is not observed in this period, this medical event is not a symptom anymore. In some cases such as infections, only medical events that happen during the incubation period are indicative of infection.

Considering all these cases, we compute a rich set of time variables ψ_i from the raw timestamp t_i in the following ways.

- Delta time (δ): This represents the time difference between documents $\mathbf{d}_i$ and $\mathbf{d}_{i-1}$ and computed as $\mathbf{t}_i - \mathbf{t}_{i-1}$.
- Ordinal time (γ): This represents the time passed since the admission of the patient and for $\mathbf{d}_i$, computed as $\mathbf{t}_i - \mathbf{t}_0$,
- Time to index event (ξ): This represents the remaining time to the index

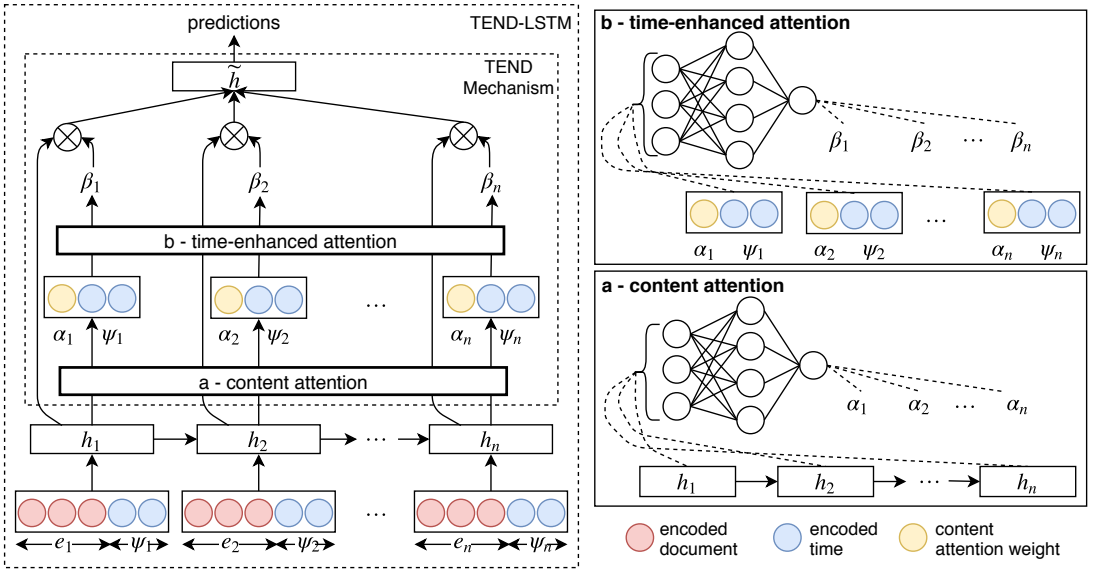


Figure 4.3: Proposed TEND-LSTM model and its components.

event. For a disease prediction problem, index event refers to the diagnosis of the disease or termination of the hospital stay. For $\mathbf{d}_i$, it is computed as $t_{L^{(n)}} - t_i$.

All time functions are computed both in days and in hours, creating six time variables for each document in total.

$$\psi_i = \left[\delta_i^h, \delta_i^d, \gamma_i^h, \gamma_i^d, \xi_i^h, \xi_i^d\right] \tag{4.2}$$

We then concatenate this time vector ψ_i with the mean word embedding vector $\mathbf{e}_i$ to create the time enhanced document vector:

$$\tau_i = [\mathbf{e}_i, \psi_i] \tag{4.3}$$

4.2.3 Long Short-Term Memory Recurrent Neural Network

After building the time-enhanced document level representations, $\tau_i, i = 1, \cdots, L$, we train an LSTM network as described in section 2.1.1 by inputting these vectors. The LSTM reads the input document vectors, then calculates a sequence of hidden states:

$$h_i = \text{LSTM}(\tau_i), i \in [1, L] \tag{4.4}$$

4.2.4 TEND: Time Enhanced Dual Attention

Our proposed TEND mechanism is composed of two hierarchical layers of attention, the first layer is called the content attention and the second layer is called the time-enhanced attention (see Figure 4.3).

Content attention In this layer, a set of attention weights is computed based on the content of the documents. To compute these weights, the output of the LSTM from Equation 4.4 is fed into a multilayer perceptron as follows:

$$u_i = \tanh(W_c h_i + b_c) \tag{4.5}$$

$$\alpha_i = \frac{\exp(u_i^\top u_c)}{\sum_t \exp(u_i^\top u_c)} \tag{4.6}$$

Here, we compute u_i as a hidden representation of h_i. We also define a context vector u_c, which is randomly initialized and jointly learned. The importance of each document is measured by the similarity of u_i with the context vector u_c. We compute normalized importance weights α_i through a softmax function as shown in Equation 4.6.

Time-enhanced attention In the time-enhanced attention layer, we compute a second set of attention weights. To this end, we concatenate the output of the content attention layer with the time vector. Using a similar logic with the previous layer, time-informed dual attention weights β are computed.

$$c_i = [\alpha_i, \psi_i] \tag{4.7}$$

$$v_i = \tanh(W_d c_i + b_d) \tag{4.8}$$

$$\beta_i = \frac{\exp(v_i^\top v_d)}{\sum_t \exp(v_i^\top v_d)} \tag{4.9}$$

The patient level representation is modeled by weighted averages of the hidden vectors of documents with these new attention weights.

$$\tilde{h} = \sum_i \beta_i h_i \tag{4.10}$$

The purpose of the dual attention layer is to integrate time and content attention. The Integration function is task-specifically learned through examples.

Table 4.2: Dataset Statistics

Statistics		Dataset					
		CDIF	MRSA	E. Coli	Enterococcus	K. Pneumoniae	Mortality
# Notes / Patient	Mean	32	14	22	28	34	30
	Median	14	7	9	13	13	19
# Words / Note	Mean	339	379	284	267	276	273
	Median	207	218	187	185	183	189
Total # Notes		66,486	35,332	82,098	129,926	71,530	299,256
Total # Patients		2,070	2,480	3,626	4,568	2,077	10,000

4.2.5 Patient Classification

The resulting patient vector $\tilde{h}$ can be seen as a dense embedding for the patient. The prediction layer takes the patient vector $\tilde{h}$ and outputs a binary label. We use cross-entropy as the objective function where $\hat{y}$ is the prediction and y is the true label.

$$L = -\big(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})\big) \tag{4.11}$$

4.3 Experimental Evaluation

4.3.1 Datasets

MIMIC III Dataset For our experimental evaluation, we use the MIMIC III Database [46], a publicly available critical care data set collected from the Beth Israel Deaconess Medical Center Intensive Care Unit (ICU) between 2001 and 2012. The database consists of all unstructured notes taken by the caregivers including nursing progress notes, discharge summaries, etc. among other information for $45,000$ patients. MIMIC is a benchmark dataset frequently used in clinical machine learning studies [9, 22, 37, 84, 90]. We extract patient cohorts from MIMIC for six classification tasks.

Cohort extraction from MIMIC Five of the tasks are Health-care Associated Infections (HAIs) often caused by antibiotic-resistant bacteria. To identify cohorts in the MIMIC III dataset, we use the microbiology test associated with the related

bacteria found in the *Microbiology Events* table. We use the microbiology test as opposed to the ICD9 code to extract the time of diagnosis. Of the resulting patient set, we use all who have note events. These patients form the positive population. We randomly sample an equal number of negative patients to create a balanced dataset. The sixth prediction task is in-hospital mortality prediction which is widely studied by clinical machine learning researchers. Health-care professionals must assess the mortality risk of an in-patient as they make care decisions.

We only use data up to a day before diagnosis for infections and a day before patient's death for mortality prediction to assure no information regarding confirmation or treatment of the condition is included. For negative patient sets, half-way of their data is used as usually done in prior work [90, 109]. Below, we provide details regarding each prediction task.

- **Clostridium Difficile Infection**: Clostridium difficile infection (CDIF) is a common HAI resulting in gastrointestinal illness with high morbidity and mortality rates. To identify CDIF patients, we use the microbiology test associated with the organism *80139-Clostridium Difficile*.

- **MRSA Infection**: Methicillin-resistant Staphylococcus aureus (MRSA), an antibiotic-resistant bacteria, is a common cause of serious HAIs. MRSA infections may result in serious complications including sepsis and death. We use the microbiology test associated with the organism *80293-MRSA* to extract MRSA cohort.

- **Escherichia Coli**: E. Coli are a diverse group of bacteria that can cause diarrhea, urinary tract infections, respiratory illness, and pneumonia. To identify E. coli patients, we use the microbiology test associated with the organism *80002-Escherichia Coli*.

- **Enterococcus Sp**: Enterococcus Sp is the leading cause of nosocomial bacteremia, surgical wound infection, and urinary tract infection. To identify Enterococcus Sp patients, we use the microbiology test associated with the organism *80053-Enterococcus Sp*.

- **Klebsiella Pneumoniae**: Klebsiella is a bacteria that can cause different HAIs, including pneumonia. To identify K. Pneumoniae patients, we use the

microbiology test associated with the organism *80004-Klebsiella Pneumoniae*.

- **In-hospital Mortality Prediction**: Detecting high-death-risk patients is a benchmark task in clinical predictive modeling research [45, 75]. We use *hospital_expire_flag* from the *Admissions* table to extract mortality-positive patient cohort.

We detail statistics for each dataset in Table 4.2.

4.3.2 Implementation Details

Data Preparation There are notes from 15 categories in MIMIC. These categories are Case Management, Consult, ECG, Echo, Discharge summary, General Nursing, Nursing/other, Nutrition, Pharmacy, Physician, Radiology, Rehab Services, Respiratory, Social Work. We use notes from all categories except for discharge summaries, as they are created at the end of the hospital stay and we only use EHR data before the diagnosis of the condition. We also note that the current literature is heavily based on discharge summaries which are semi-structured, unlike the other categories.

We lowercase and remove punctuation from the clinical notes. MIMIC dataset contains de-identified data so sensitive information, such as doctor or patient name, is deducted and put in square brackets. (e.g., Attending:[**First Name3 (LF) 1**]). We also remove these parts from the notes. The creation-time of each note is used to extract time features. Lastly, *age* is extracted from the 'patients' table.

Model Training For all 6 datasets, we learn local word embeddings using all notes that belong to patients from the training set of the relevant cohort. We use the skip-gram architecture of Word2vec [67] to learn vector representation of words. We set skip window size to 1, number of skips to 2, and number of negative examples to sample to 64. Our embedding size is 32. We implement our model with Tensorflow. Based on the mean/median number of notes, we used 20 notes per patient. In the case that a patient has fewer than 20 documents, we pad their sequences up to 20 with vectors of 0's. If a patient has more notes, earlier notes are discarded to keep only last 20. We use the Adam optimizer. Exponential decay learning rate schedule is used with an initial rate of 0.01. Model performance is measured by accuracy.

Table 4.3: Performance comparison of baseline methods and the proposed method

Method	Dataset					
	CDIF	MRSA	E. Coli	Enterococcus	K. Pneumoniae	Mortality
LSTM	56.20 ± 0.71	65.88 ± 1.34	62.98 ± 1.06	59.49 ± 2.72	60.14 ± 1.34	65.94 ± 2.98
Time features	57.18 ± 1.44	71.66 ± 1.04	66.80 ± 0.95	62.46 ± 0.81	62.00 ± 0.94	70.42 ± 0.98
Content att.	60.67 ± 1.00	70.53 ± 0.67	65.24 ± 1.17	64.31 ± 0.58	64.35 ± 0.79	72.73 ± 0.64
Time att.	61.33 ± 0.76	70.33 ± 0.67	66.24 ± 0.79	64.86 ± 0.41	64.62 ± 0.87	72.26 ± 0.66
Reciprocal time att. [14]	60.04 ± 1.46	70.93 ± 0.59	66.21 ± 0.32	65.22 ± 0.75	64.78 ± 0.47	72.24 ± 0.42
Time-decay att. [96]	61.16 ± 0.63	70.74 ± 0.46	66.05 ± 0.22	65.09 ± 0.91	64.53 ± 0.99	72.10 ± 0.20
Time+Content att. [14]	60.85 ± 0.43	70.68 ± 0.42	67.06 ± 0.42	65.28 ± 0.84	65.03 ± 0.26	72.81 ± 0.36
T-LSTM [9]	59.9 ± 1.77	72.84 ± 1.50	68.67 ± 0.63	65.08 ± 1.36	63.84 ± 0.85	75.78 ± 0.63
TEND-LSTM	$\mathbf{62.30} \pm 0.71$	$\mathbf{76.52} \pm 1.19$	$\mathbf{69.87} \pm 0.59$	$\mathbf{68.35} \pm 0.48$	$\mathbf{65.72} \pm 0.28$	$\mathbf{76.24} \pm 0.33$

Table 4.4: Effect of individual time variables

Method	Dataset					
	CDIF	MRSA	E. Coli	Enterococcus	K. Pneumoniae	Mortality
No time	56.20 ± 0.71	65.88 ± 1.34	62.98 ± 1.06	59.49 ± 2.72	60.14 ± 1.34	65.94 ± 2.98
Ordinal hour	57.11 ± 0.93	64.63 ± 2.60	64.46 ± 2.34	60.64 ± 3.84	59.71 ± 1.36	68.26 ± 0.87
Delta hour	55.84 ± 0.92	62.43 ± 3.35	63.81 ± 1.28	58.63 ± 3.46	60.00 ± 4.42	68.63 ± 1.00
Hours to index	57.25 ± 0.88	66.10 ± 1.53	64.70 ± 1.35	58.98 ± 2.13	60.74 ± 1.28	69.50 ± 1.28
Ordinal day	59.11 ± 2.42	70.25 ± 1.20	63.85 ± 1.14	61.51 ± 0.99	60.70 ± 0.59	69.75 ± 1.52
Delta day	$\mathbf{59.76} \pm 1.07$	69.72 ± 0.72	65.34 ± 0.73	62.34 ± 0.98	$\mathbf{62.35} \pm 0.6$	68.87 ± 1.24
Days to index	59.66 ± 1.07	70.17 ± 0.87	63.42 ± 0.50	58.61 ± 1.10	59.81 ± 1.62	67.84 ± 0.81
All variables	57.18 ± 1.44	$\mathbf{71.66} \pm 1.04$	$\mathbf{66.80} \pm 0.95$	$\mathbf{62.46} \pm 0.81$	62.00 ± 0.94	$\mathbf{70.42} \pm 0.98$

4.3.3 Performance of TEND-LSTM Model

To evaluate our TEND-LSTM model, we design four baseline methods.

- **LSTM**: We use only the sequential note level representations to feed into an LSTM model. No time information is used in this model.
- **LSTM w/Content attention**: We add only the content attention on top of the previous model. No time information is used.
- **LSTM w/Time features**: We first fuse document level summary with time-based features into one. Combined representation is then fed into an LSTM model. No attention mechanism is used.

- **LSTM w/Time attention**: This model uses the time information in the form of an attention mechanism on top of LSTM. Computed input time vector is fed through a fully connected layer to learn the time attention weights.

We also compare TEND-LSTM with state-of-the-art time-aware models. In all the methods described below, the time input to the model is a scalar (time difference between instances) instead of a vector. Therefore, in these models, we experiment with delta day and delta hour and report the accuracy for whichever is higher.

- **Reciprocal time attention**: This method is proposed by [14]. Reciprocal of the time difference between instances serves as the time attention weight.

- **Time-decay attention**: Convex, concave, and linear time functions are combined as proposed in [96] and resulting weights are used for attention.

- **Time + Content attention**: We first compute content attention. Then for each note, the content attention weight is multiplied by the time variable. This method is proposed by [14].

- **T-LSTM** Time aware LSTM is proposed in [9]. This model first decomposes memory cell in LSTM into long-term memory and short-term memory, then applies time decay factor to discount the short term memory, and finally calculates the new memory by combining the long-term memory and a discounted short-term memory.

The results are presented in Table 4.3. TEND-LSTM outperforms all eight alternate methods for all six classification tasks as highlighted in bold. We observe that using time information, regardless of the method and the dataset, improves classification performance. Lowest accuracy belongs to vanilla LSTM model for all prediction tasks. Using time in the form of attention performs better than using it as simple features (Significantly better for 4 out 5 tasks). For all infection prediction tasks, Time attention performs better than Content attention, proving that timing of the notes may be more indicative of a condition then it's content. For mortality prediction, however, the opposite is true.

4.3.4 Effectiveness of Individual Time Variables as Features

We also analyze the performance improvements stemming from including each time variable. To this end, we compare the LSTM model where no time information is used with including every time variable one at a time and all time variables added at once. Table 4.4 presents the results of these experiments. For four out of six tasks, using all time variables together outperforms all individual time variables. Among the time variables, delta day (δ^d) and ordinal day (γ^d) achieve the best performance. This proves that different time reference points are more representative depending on the medical condition that is being predicted. For four out of six tasks, day-based variables (ordinal day, delta day, and days to index) perform better than hour-based variables (ordinal hour, delta hour, and hours to index).

We would like to note that, time to index feature cannot be used in a real-life setting as this information would be unknown. However, as it is used in the literature [16], we wanted to explore this feature and asses its predictive power. We observe that, this time feature does not achieve significantly better results compared to other representations.

4.3.5 Effect of Individual Time Variables in the Attention Layer: Case Study on CDIF

We experiment with every time variable and their effectiveness when used for the time attention. For this, using the document level representations, we run two different experiments: 1) An LSTM with content attention, 2) An LSTM with time attention where individual time variables are used as the time attention. Results for CDIF are presented in Table 4.5. These results shed some light on the characteristics of this specific infection. Clostridium difficile infection is an acute bacterial infection with a short incubation period (3-7 days, [90]). Previous works intuitively show that, EHR data closer to the index event (diagnosis of the infection) is more predictive than the data from earlier time spans [90]. Using *ordinal* variables at the time attention level puts more emphasis on the notes closer to the diagnosis of the infection (See Figure 4.4). In other words, the closer a

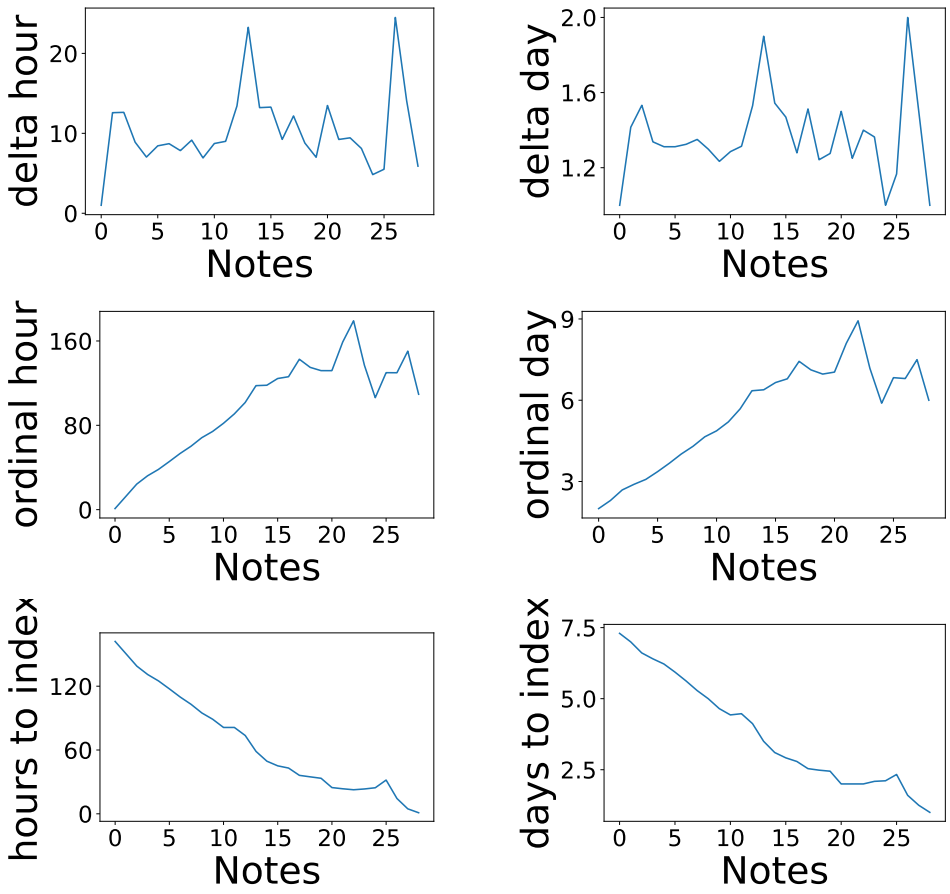


Figure 4.4: Average distribution of the time variables across all patients. Ordinal time variables follow an increasing trend while time-to-index variables behave the opposite way. Delta variables are a mixture of both.

note to the diagnosis, the more attention it gets. Using *delta* variables puts on average equal attention to all notes, if there is not a big time difference between two consecutive notes. *Time to index event* variables work the opposite of *ordinal* variables, that is, earlier notes get higher attention from the model. For CDIF, using only ordinal variables at the time attention layer gets better results compared to the content attention and other variables. Similarly, time-to-index variables achieve worse results compared to content attention. These results are consistent with and validated by the characteristics of the infection.

Table 4.5: Effect of individual time variables in the attention layer for CDIF

Model	Accuracy
Content attention	
Content att.	60.67 ± 1.00
Time attention	
Ordinal hour	61.14 ± 0.80
Ordinal day	60.94 ± 0.82
Delta hour	60.91 ± 0.36
Delta day	59.56 ± 1.18
Hours to index	60.40 ± 1.52
Days to index	60.03 ± 0.45

4.4 Summary

For this task, we design a novel attention mechanism, TEND: Time Enhanced Dual Attention, and an end-to-end deep network architecture utilizing this attention mechanism, TEND-LSTM. Our TEND-LSTM model is effective for sequential document classification tasks where the input documents have associated timestamps. The TEND deep network is comprised of two attention layers, the first layer is to learn content based attention for the document sequence and the second layer is to learn a task-specific combination of content and time. We evaluate the performance of the TEND-LSTM model using six real-world clinical note datasets. Empirical results show that TEND-LSTM outperforms strong baselines and state-of-the-art methods.

This task is resulted in the following publication:

- C. Sen, T. Hartvigsen, X. Kong, E. Rundensteiner, "Learning Temporal Relevance in Longitudinal Medical Notes", In 2019 IEEE International Conference on Big Data (Big Data), pp. 2474-2483. IEEE, 2019.

Chapter 5

Human vs Machine Attention

5.1 Defining Interpretability

The machine learning field has seen unprecedented progress in recent years due to the success of many models, and especially the deep learning models. Many applications fueled by machine learning and deep learning have paved their way into everyday life, such as recommendations by Netflix, speech recognition by Amazon Alexa or Apple Siri, and neural machine translation by Google. Despite their success, especially deep learning suffers from some limitations. The most prominent drawback of these deep learning models is the lack of transparency in their actions. This leaves end-users with little understanding of how these models make particular decisions.

As machine learning models being used in critical areas such as medicine, the criminal justice system, and financial markets, the inability of humans to understand these models is problematic. Explainable or interpretable models can solve this problem. However, few articulate precisely what interpretability means, leaving the term "interpretability" ill-defined. Hence, the claims regarding the interpretability of various models may exhibit a quasi-scientific character [58]. In the following sections, I summarize the recent research efforts to define the concept of interpretability.

5.1.1 Properties of Interpretability

One of the first efforts for defining interpretability is by Lipton [58]. According to Lipton, the properties of interpretable models are two-fold.

The first one is **transparency**. Transparency is informally defined as the opposite of opacity or blackboxness. It indicates some understanding of the mechanism that the model works. Transparency can be considered at three levels.

1. *Simulatability* - at the level of the entire model: We can call a model transparent if a person can "contemplate" the entire model at once. According to this definition, an interpretable model is a simple model. This corresponds to the common claim that sparse linear models (e.g., lasso regression) are more interpretable than dense linear models.

2. *Decomposability* - at the level of individual components (e.g., parameters): Another way of defining transparency might be that each part of the model (e.g., input, parameter) reveals an intuitive explanation. For example, each node in a decision tree might correspond to a description.

3. *Algorithmic transparency* - at the level of the training algorithm: Deep learning models lack algorithmic transparency. Even though being empirically powerful, we don't understand how deep networks work, and cannot guarantee beforehand that they will work on new problems.

The second property is referred to as **post-hoc interpretability**. This concept refers to extracting information from learned models. This type of interpretability is not informative about how a model works. However, it can still provide useful information for end-users of machine learning. Post-hoc interpretability can be achieved in three ways.

1. *Natural language explanations:* Humans justify decisions verbally. So the idea here is to train a model to generate predictions and a separate model to generate explanations.

2. *Visualizations of learned representations or models:* Another approach is to create visualizations for a qualitative measure of interpretability. For example,

one traditional approach is to visualize high-dimensional representations with dimensionality reduction methods such as t-SNE.

3. *Explanations by example:* Similar examples classified by the models can be used as a means of interpretability (e.g., this tumor is classified as malignant, because it looks a lot like these other tumors).

5.1.2 Evaluating Interpretability

Another prominent research paper on interpretability is by Doshi-Velez et al. [25]. Their main goal is to define interpretability, and more importantly, laying out a methodology for evaluating it.

According to this paper, "interpret" means to explain or to present in understandable terms. In the context of machine learning, interpretability is defined as the ability to explain or to present in understandable terms to a human.

In this work, the authors categorize ways of evaluating interpretability into two.

1. By using real humans and real tasks or simplified tasks: This approach aims to evaluate interpretability with respect to an application. "Application-grounded evaluation" requires conducting human experiments using the application-at-hand. For example, if the application-at-hand is diagnosing patients with a particular disease, this way of evaluation requires doctors to perform the diagnoses.

2. By using proxy tasks instead of humans: This approach is termed as "Functionally-grounded evaluation", and it does not require human experiments. Instead, it uses a formal definition of interpretability as a proxy for explanation quality. In this approach, a researcher should first claim that some model class, e.g., sparse linear models, are interpretable and then present algorithms to optimize within that class. Such experiments are appealing because human-subject experiments require time and costs.

5.1.3 Achieving Interpretability in Deep Learning

In their recent paper, [26] defines interpretability and puts forth a systematic classification of methods for achieving interpretability in machine learning.

According to [26], interpretability is the ability of models to explain or to present their behaviors in understandable terms to humans. A good explanation should be meaningful to users and avoid being influenced by artifacts. According to this definition, interpretable machine/deep learning is a multidisciplinary concept, and it requires efforts from many fields such as computer science, social science, and human-computer interaction. Only this way, real user-oriented and human-friendly explanations can be designed.

[26] suggest a categorization of interpretable methods based on the time when the interpretability is obtained: **intrinsic interpretability** and **post-hoc interpretability**.

Intrinsic interpretability refers to building self-explanatory models where interpretability is directly incorporated into the model structure. Examples of these types of models include rule-based models, decision trees, linear models, and attention models. These models could provide accurate explanations but may sacrifice prediction performance.

On the other hand, post-hoc interpretability involves building a secondary model specifically for providing explanations for the main model. The primary difference between these two types comes from the trade-off between model accuracy and explanation fidelity. These models are limited in their approximate nature while keeping the underlying model accuracy intact.

5.2 Motivation

When humans perform a cognitive task, they pay varying amounts of attention to different parts of the task. For example, when reading a text, they pay more attention to specific words while skipping others [48]. Inspired by this observation, a recent trend in deep learning is to build computational models of attention [5]. Such neural attention mechanisms allow neural network models to adjust their focus to specific parts of the input data, improving model performance and adding interpretability.

Attention-based models have become the architectures of choice for a vast number of NLP tasks including, but not limited to, language modeling [20], machine translation [5], document classification [114], and question answering [54, 98]. While

attention mechanisms have been said to add interpretability since their introduction [5], the investigation of whether this claim is correct has only just recently become a topic of high-interest [68, 92, 104]. If attention mechanisms indeed offer a more in-depth understanding of a model's inner-workings, application areas from model debugging to architecture selection would benefit greatly from profound insights into the internals of attention-based neural models.

Recently, [41], [108], and [92] proposed three distinct approaches for evaluating the explainability of attention. [41] base their work on the premise that explainable attention scores should be unique for a given prediction as well as consistent with other feature-importance measures. This prompts their conclusion that attention is not explanation. Based on similar experiments on alternative attention scores, [92] conclude that attention does not necessarily correspond to the importance of inputs. In contrast, [108] find that attention learns a meaningful relationship between input tokens and model predictions, which cannot be easily 'hacked' adversarially.

While these works ask valuable questions, they embrace model-driven approaches for manipulating the attention weights and thereafter evaluate the post-hoc explainability of the generated machine attention. In other words, they overlook the human factor in the evaluation process – which should be integral in assessing the plausibility of the generated explanations [85].

In this work, we adopt a novel approach to attention explainability from a human-centered perspective and, in particular, investigate to what degree machine attention *mimics* human behavior. More precisely, we are interested in the following research question: *Do neural networks with attention mechanisms attend to the same parts of the text as humans?* To this end, we first collect a large dataset of human-attention maps and then compare the validated human attention with a variety of machine attention mechanisms for text classification.

Figure 6.1 displays examples of human and machine-generated attention for classifying a restaurant review's overall rating. Our goal is to quantify the similarity between human attention and machine-generated attention scores. Measuring this similarity is non-trivial and is not appropriately captured by an existing similarity metric (*e.g.*, Euclidean) between two vectors for the following reasons. A binary human attention vector does not solely denote which tokens are given higher importance but also implies information about the underlying grammatical

structure and linguistic construction. For example, whether or not adjectives tend to be high-importance is encoded in the attention weights as well. Further, it is well known that human attention is itself subjective: given the same text and task, human annotators may not always agree on which words are important. That is, one single human's attention should rarely be regarded as the ground-truth for attention.

Given this objective, we use crowd-sourcing to collect a large set of human attention maps. We first provide a detailed account of the iterative design process for our data collection study. We design new metrics that quantify the similarity between machine and human attention from three perspectives: *Behavioral similarity* measures the number of common words selected by human and machine discerning if neural networks with attention mechanisms attend to the same parts of the text as humans. Humans associate certain lexical categories (*e.g.*, adjectives) with a sentiment more heavily. *Lexical (grammatical) similarity* identifies if machine attention favors similar lexical categories with humans. A high lexical similarity shows that the attention mechanism learns similar language patterns with humans. *Context-dependency* quanitifies sentiment polarity of word selections.

We then employ these metrics to compare attention maps from a variety of attention-based Recurrent Neural Networks (RNN). We find that bi-Directional RNNs with additive attention demonstrate strong similarities to human attention for all three metrics. In contrast, uni-directional RNNs with attention differ from human attention significantly. Finally, as the text length increases, and with it, the prediction task becomes more difficult, both the accuracy of the models and similarity between human and machine decrease.

Our contributions are as follows:

- We conduct a large-scale collection of 15,000 human attention maps as a companion to the publicly-available Yelp Review dataset. Our collected Yelp-HAT (Human ATtention) dataset is publicly available as a valuable resource to the NLP community.

- We develop rich metrics for comparing human and machine attention maps for text. Our new metrics cover three complementary perspectives: *behavioral similarity*, *lexical similarity*, and *context-dependency*.

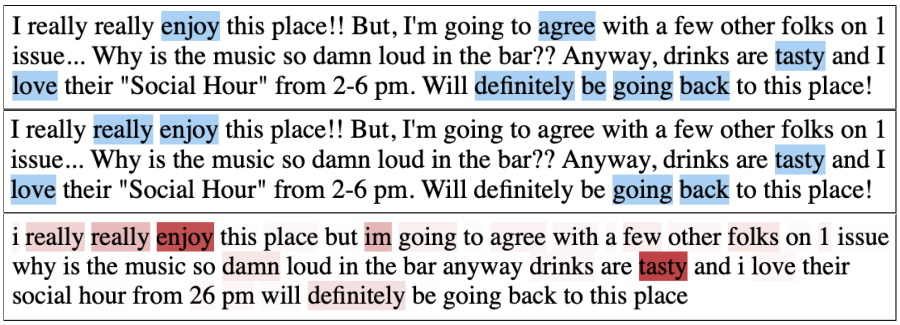


Figure 5.1: Examples of binary human attention (top two figures) and continuous machine attention (bottom figure).

- We conduct the first in-depth assessment comparing human versus machine attention maps, with the latter generated by a variety of state-of-the-art soft and hard attention.
- We show that when used with bidirectional architectures, attention can be interpreted as human-like explanations for model predictions. However, as text length increases, machine attention resembles human attention less.

5.3 Preliminaries and Definitions

In this section, we define the concepts of Human Attention Map and Machine Attention Map.

Definition 5.3.1. Attention Map. An *Attention Map (AM)* is a vector where each entry in sequence is associated with a word in the corresponding position of the associated text. The value of the entry indicates the level of attention the corresponding word receives with respect to a classification task.

Definition 5.3.2. Human Attention Map. A Human Attention Map (HAM) is a binary attention map produced by a human, where each entry with a set-bit indicates that the corresponding word receives high attention.

Definition 5.3.3. Machine Attention Map. A Machine Attention Map (MAM) is an attention map generated by a neural network model. If computed through *soft-attention*, a MAM corresponds to an AM of continuous values, that capture

a probability distribution over the words. If computed through *hard-attention*, a MAM is a binary AM.

We now introduce the application of aggregation operators to coalesce HAMs by multiple annotators into aggregated HAMs.

Definition 5.3.4. Consensus Attention Map. If multiple HAMs exist for the same text, a Consensus Attention Map (CAM) is computed through a bitwise AND operation of the HAMs.

Definition 5.3.5. Super Attention Map. If multiple HAMs exist for the same text, a Super Attention Map (SAM) is computed by a bitwise OR operation of the HAMs.

5.4 HAM Collection by Crowd-sourcing

We collect human attention maps for the Yelp dataset[1] on the classification task of rating a review as positive or negative on Amazon Mechanical Turk. Participants are asked to complete two tasks: 1) Identify the sentiment of the review as positive, negative, or neither, and 2) Highlight the words that are indicative of the chosen sentiment. Our interface used for data collection is in Figure 5.2.

5.4.1 Preliminary investigation of the quality of human annotations.

First, we conduct a series of data collection studies on two subsets of the Yelp dataset. Both subsets consist of 50 randomly-selected reviews from the *Restaurant* category. The first subset contains reviews with exactly 50 words, while the second contains reviews with exactly 100 words. For each review, human annotation is collected from two unique users.

We explore the quality of data we can collect on Mechanical Turk, as it encourages users to complete their tasks as quickly as possible since the number of completed tasks determines their income. This may lower the quality of collected

[1] https://www.yelp.com/dataset/challenge

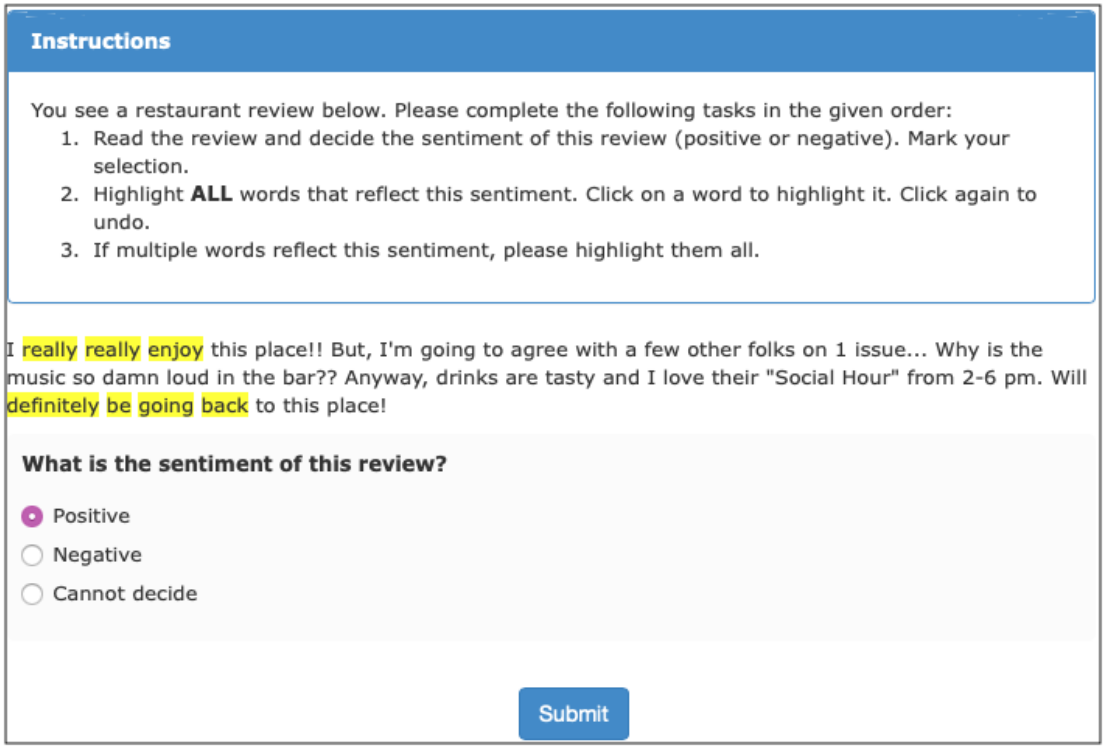


Figure 5.2: User interface we used for data collection on Amazon Mechanical Turk.

data since users may not select all relevant words, instead opting for the few most obvious ones, or they may choose words randomly.

Based on our preliminary investigations, we observe that both the average time users spend on the task (44 vs. 70 seconds) and the average number of words selected per review (9 vs. 13 words) increase as the number of words in the review increases from 50 to 100. This suggests that users do not choose words randomly; instead, they make an informed decision. We also visually examine the collected human attention maps and confirm that subjects make meaningful selections.

5.4.2 Pilot study assessing two design choices for data collection.

Next, we design another pilot study to understand how humans perform the cognitive task of classifying a text and selecting the particular words that led to this decision. In this study, we ask eight participants to perform the same task

while adhering to one of two strategies. The first strategy, the *read-first* design, involves reading the review first, deciding on the sentiment, then rereading the review, this time to highlight the relevant words. The second strategy, the *free-style* design, gives participants the freedom to choose the relevant words as they read the review to determine the sentiment. Each participant is asked to complete two tasks to experience both strategies. Half of the participants first work with the *read-first* design followed by the *free-style* design while the other half work in the reverse order. After completing the tasks, we ask the participants which strategy they find more natural in a post-task questionnaire.

5.4.3 Findings from the pilot study.

Out of eight participants, half of them find it more useful reading the review first then deciding on the words whereas the other half indicated the opposite. We then evaluate the collected data from three perspectives to decide which design is most suitable for our purposes.

We first examine the agreement between participants adhering to a particular strategy. This involves calculating the percentage of participants that mutually select the same phrase. We find that participant agreement is higher (73%) when the participants are forced to read the review before making any selections compared to using the free-style design (69%). Next, we investigate how similar the results are to the ground truth we defined for each review. The read-first design achieves better performance (3.30) compared to the free-style design (3.10). Our final criterion involves examining the amount of noise in the data (i.e., selections which deviate from the chosen sentiment). Only one review exhibits this situation where the review is clearly positive; however, it also contains a negative-opinion sentence. We observe that the read-first design reduces this cross-sentiment noise (1 vs. 0.5 scores).

5.4.4 Data collection protocol for the main study.

Based on conclusions from the pilot studies, the *read-first* design is adopted to conduct the main data collection for $5,000$ reviews on Amazon Mechanical Turk. For this study, three different subjects annotated each review, resulting in a total

of 15,000 human attention maps. The resulting Yelp Human Attention Dataset (YELP-HAT) is publicly available [2].

5.4.5 Analysis and Insights About HAMs

5.4.5.1 Factors that affect human accuracy.

Some reviews contain a mixture of opinions, even though the reviewer felt strongly positive or negative about the restaurant. For example, consider the following review: *"Nothing to write home about, the chicken seems microwaved and the appetizers are meh. ... If your [sic] looking for a quick oriental fix I'd say go for it.. otherwise look elsewhere."* This review is labeled as *negative*, *positive*, and *neither*. The annotator who assigned it to the positive class selected the words "go for it" while the annotator who assigned it to the negative class selected the words "otherwise look elsewhere". This type of "mixed review" is the principal reason for discrepancies in classifications by the human annotators. The nature of crowd-sourcing also causes such inconsistencies as not all annotators provide reviews of equal quality.

5.4.5.2 Ambiguity in human attention.

Intuitively, human attention is highly subjective. Some common patterns across annotators lead to differences in human annotations. A common behavior is to select *keywords* that indicate a sentiment. Another typical action is to select *entire sentences* if the sentence expresses an opinion.

Some reviews include subjective phrases that people interpret differently with regard to sentiment-polarity. For instance, "I come here often" can be construed as a favorable opinion. However, some people find it neutral. In some cases, an overwhelmingly-positive review incorporates a negative remark (or vice versa). In these cases, some people select all pieces of evidence of any sentiment, whereas others only choose words that indicate the prevailing sentiment.

[2] http://davis.wpi.edu/dsrg/PROJECTS/YELPHAT/index.html

5.5 Attention Map Similarity Framework

We quantify the similarity between HAMs and MAMs through our similarity framework that contains three new metrics as described in this section.

5.5.1 Overlap in Word Selections

For two attention mechanisms to be similar, they must put attention on the same parts of the text. Thus, we first define a metric for quantifying the overlap in the words selected by human annotators and by deep learning models.

Definition 5.5.1. Behavioral Similarity. Given a collection of attention maps HAM_D and MAM_D for a text dataset D, behavioral similarity between human (H) and machine (M) corresponds to the average pair-wise similarity between each $(\text{HAM}_i, \text{MAM}_i)$ vector pair $\forall i \in D$ as defined below:

$$\text{PairwiseSim}_i = AUC(\text{HAM}_i, \text{MAM}_i)$$

$$\text{BehavioralSim}(M, H) = \frac{1}{|D|} \sum_i (\text{PairwiseSim}_i)$$

where $|D|$ is the number of reviews in the dataset $\mathcal{D}$. Word Similarity metric first computes an AUC score between a human attention vector and a machine attention vector for each pair. Average of these pairwise similarities over all map pairs corresponds to the overall similarity between human and machine. Intuitively, this corresponds to adopting the human attention vector as binary ground truth. That is, it measures how similar the machine-generated continuous vector is to this ground truth. AUC is between 0 and 1 with .5 representing no similarity, and 1 the perfect similarity.

5.5.2 Distribution over Lexical Categories

Previous work has found that lexical indicators of sentiment are commonly associated with syntactic categories such as adjective, adverb, noun, and verb [66]. We define the following lexical similarity metric to test if human and machine adopt similar behaviors in terms favoring certain lexical categories.

Definition 5.5.2. Lexical Similarity. Given a collection of attention maps HAM_D and MAM_D for a text dataset D, Lexical Similarity (LS) between human (H) and machine (M) over D is computed:

$$\text{LS}(M, H) = \text{corr}(\text{dist}(\text{words}_H), \text{dist}(\text{words}_M))$$

where words_H is a list of all selected words in all reviews of $\mathcal{D}$ by human, words_M is a list of all selected words in all reviews of $\mathcal{D}$ by machine, dist() is a function that computes the distribution of a word list over a tagset (e.g., nouns, verbs, etc.). The complete tagset list we used can be found online [3]. After computing two distributions, the corr() function computes the correlation between them. In our experiments, we adopt Pearson Correlation. If MAM is continuous, selected words by M corresponds to k words with the highest attention scores, where k is the number of words selected by human for that text.

Using a *random attention* R as a baseline where the most important k words are selected randomly, we then compute an *Adjusted Lexical Similarity* which is between 0 and 1 as follows.

$$\text{AdjustedLS} = \frac{\text{LS}(M, H) - \text{LS}(R, H)}{1 - \text{LS}(R, H)}$$

5.5.3 Context-dependency of Sentimental Polarity

When deciding the sentiment of a review, human subjects may consider positive sentiment words in a negative review and vice versa. We define context-dependency as the selection frequency of an opposite-sentiment word (e.g., "good" selected in a negative review). To assess how context-dependant human and machine attentions are, we compute cross-sentiment selections rates.

Definition 5.5.3. Cross-sentiment selection rate (CSSR). Assume we have a collection of attention maps AM_D for a dataset D, ground truth for overall sentiment Y for each review in D ($y_i \in \{0, 1\}$), and a list of positive words $\mathcal{P}$

[3] https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

and negative words $\mathcal{N}$ in the English language. CSSR denotes the ratio of selected words from the opposite sentiment.

$$p\_words = get\_words(HAM_D, Y = 1)$$

$$n\_words = get\_words(HAM_D, Y = 0)$$

$$CSSR_p = \frac{|p\_words \cap \mathcal{N}|}{|p\_words \cap \mathcal{P}|}$$

$$CSSR_n = \frac{|n\_words \cap \mathcal{P}|}{|n\_words \cap \mathcal{N}|}$$

get_words() function returns a list of attention-receiving words where $\text{HAM}_{ij} = 1, \forall i, j$ for the entire set of HAM_D, for positive-sentiment reviews ($Y = 1$) and negative-sentiment reviews ($Y = 0$) separately. A list of words with positive and negative connotations, $\mathcal{P}$ and $\mathcal{N}$, are obtained from [40]. CSSR_p (positive) and CSSR_n (negative) is then computed as the ratio of the number of cross-sentiment words over the number of same-sentiment words. A high CSSR means many words from the opposite sentiment are selected. This metric provides insights about how similar human and machine attentions are with regard to their context-dependant behaviour. With the help of these pre-defined word lists, negative and positive word counts are calculated for negative and positive reviews. Contrastive word selections indicate how context-dependent is the attention mechanism.

5.6 Is Machine Attention Similar to Human Attention?

5.6.1 Generating Machine Attention Maps

The Yelp dataset contains reviews and their rating scores between 0 and 5 (stars). This rating score corresponds to the ground truth for the review's overall sentiment. We create a binary classification task by assigning 1 and 2-star reviews to the negative class and 4 and 5-star reviews to the positive class. We omit 3-star reviews as they may not exhibit a clear sentiment. For training neural network models, we extract balanced subsets and split them into 80% training set, 10%

validation set and 10% test sets. We then generate MAMs using the following machine learning models.

5.6.1.1 RNN with soft attention

Recurrent Neural Networks (RNN) enhanced with attention mechanisms have emerged as the state-of-the-art for NLP tasks [5, 20, 54, 114]. We implement the additive attention for many-to-one classification task as it is commonly used in the literature [5, 114] and paired it with both uni- and bi-directional RNN. In our implementation, we use LSTM memory cells.

Assuming that Γ is the recurrence function of LSTM and x_i is the embedded i-th word of T words in a review, we model our method as:

$$h_i = \Gamma(x_i, h_{i-1}), i \in [1, T] \tag{5.1}$$

$$u_i = \tanh(W h_i + b) \tag{5.2}$$

$$\alpha_i = \frac{\exp(u_i^\top u)}{\sum_t \exp(u_i^\top u)} \tag{5.3}$$

Here $h_i, i \in [1, T]$ are hidden representations, W, b, and u are trainable parameters, and $\alpha_i, i \in [1, T]$ are the attention scores for each word x_i. A context vector c_i corresponds to the weighted average of the hidden representations of words with attention weights, denoted by:

$$c_i = \sum_j \alpha_j h_j \tag{5.4}$$

Through a softmax layer, context vector c_i is then used for further classifying the input sequence.

5.6.1.2 Rationale mechanism

An alternative approach, referred to as "rationale mechanism", can be seen as a type of hard attention [7, 57]. This model consists of two main parts that are jointly learned: a generator and an encoder. The generator specifies a distribution over the input text to select candidate rationales. The encoder is used to make predictions based on the rationales. The two components are integrated and regularized in the cost function with two hyper-parameters, selection lambda, and continuity lambda,

	Accuracy		
	Yelp-50	Yelp-100	Yelp-200
Human	0.96	0.94	0.94
RNN	0.91 ± 0.006	0.90 ± 0.013	0.88 ± 0.01
biRNN	0.93 ± 0.008	0.91 ± 0.005	0.88 ± 0.02
Rationales	0.90 ± 0.004	0.85 ± 0.035	0.77 ± 0.015

Table 5.1: Test accuracy from three subsets of Yelp data.

for optimizing the representative selections. The selection lambda penalizes the number of words selected, while the continuity lambda encourages the continuity via minimizing the distances of the words chosen. Thus, a higher selection lambda tends to select fewer words as rationales, while a higher continuity lambda tends to promote phrases as meaningful rationales.

5.6.1.3 Implementation Details

For the Rationale Neural Prediction Framework, we use the Pytorch implementation[4] suggested by [57]. In this framework, the encoder is built as Convolutional Neural Network (CNN) and the generator is built as Gumbel Softmax with independent selectors. The following hyper-parameters of CNN are used as pointed out by [57]: 200 hidden dimensions, 0.1 dropout rate, 2 hidden layers, 128 batch size, 64 epochs, 0.0003 initial learning rate.

We conducted an extensive parameter search to find the optimum values for the two key hyper-parameters of the rationale model, selection-lambda, and continuity-lambda, which regularize the number and the continuity of words selected during the optimization process. For the selection lambda, we experimented with values 1, 1e-1, 1e-2, 1e-3, 1e-4, 1e-5, 1e-6, 1e-7, 1e-8, 1e-9, and 0. For the continuity lambda, we experimented with values 0 and two times of selection lambda. We observe that the performance of the rationale-based model is extremely sensitive to its hyper-parameters.

One conflicting interest with the rationale-based models is that the more words the model selects, the accuracy becomes higher. As our goal is to compare human

[4] https://github.com/yala/text_nn

Yelp-50	$\text{HAM}_1, k = 10$	$\text{HAM}_2, k = 12$	$\text{HAM}_3, k = 12$	$\text{CAM}, k = 5$	$\text{SAM}, k = 22$
HAM_2	0.73	-	-	-	-
HAM_3	0.74	0.75	-	-	-
RNN Attention	0.59 ± 0.021	0.59 ± 0.002	0.57 ± 0.012	0.59 ± 0.024	0.58 ± 0.021
Bi-RNN Attention	0.69 ± 0.004	0.70 ± 0.008	0.69 ± 0.007	0.79 ± 0.003	0.64 ± 0.008
Rationales	0.62 ± 0.014	0.62 ± 0.012	0.63 ± 0.015	0.68 ± 0.020	0.58 ± 0.010
Yelp-100	$\text{HAM}_1, k = 15$	$\text{HAM}_2, k = 16$	$\text{HAM}_3, k = 16$	$\text{CAM}, k = 6$	$\text{SAM}, k = 30$
HAM_2	0.71	-	-	-	-
HAM_3	0.73	0.74	-	-	-
RNN Attention	0.57 ± 0.009	0.58 ± 0.011	0.59 ± 0.012	0.57 ± 0.010	0.58 ± 0.008
Bi-RNN Attention	0.65 ± 0.011	0.65 ± 0.021	0.66 ± 0.021	0.73 ± 0.031	0.62 ± 0.012
Rationales	0.55 ± 0.015	0.55 ± 0.005	0.55 ± 0.010	0.59 ± 0.015	0.54 ± 0.005
Yelp-200	$\text{HAM}_1, k = 26$	$\text{HAM}_2, k = 27$	$\text{HAM}_3, k = 25$	$\text{CAM}, k = 11$	$\text{SAM}, k = 45$
HAM_2	0.70	-	-	-	-
HAM_3	0.69	0.71	-	-	-
RNN Attention	0.60 ± 0.011	0.60 ± 0.013	0.60 ± 0.014	0.60 ± 0.017	0.60 ± 0.011
Bi-RNN Attention	0.61 ± 0.015	0.61 ± 0.008	0.61 ± 0.018	0.63 ± 0.009	0.60 ± 0.008
Rationales	0.51 ± 0.013	0.52 ± 0.021	0.51 ± 0.018	0.52 ± 0.025	0.49 ± 0.019

Table 5.2: Behavioral similarity of human attention to machine on varying review length. k indicates the average number of words selected. (0.5:no similarity, 1.0:perfect similarity)

attention with machine-generated attention for model interpretability, we optimize the model not only for accuracy but also for the number of selected rationales. We aim to generate roughly an equal number of words selected by both human annotators and machine-generated rationales.

We aim to get roughly equal number of words with human annotators as opposed to optimizing for the accuracy. The goal was to observe 2 optimal experiments with less words selected and higher accuracy in the test set of every Yelp review dataset. And we would continue to tune selection lambda if no optimal results found, e.g. rationales are 1 with poor accuracy in all experiments under a dataset.

For training Attention-based models, we used the following hyper-parameters to RNN-based models. 100 hidden dimensions, 100 attention size, 0.2 dropout rate, 128 batch size, 64 epochs, 0.0001 initial learning rate.

5.6.2 Behavioral Similarity Analysis

We conduct a set of controlled experiments with the length of the review changing across experiments. First, we generate MAMs for three subsets of the Yelp dataset: reviews containing 50 words (Yelp-50), 100 words (Yelp-100) and 200 words (Yelp-200). Neural network models with attention mechanisms are trained on each of these subsets. The corresponding test set accuracies for sentiment classification of human versus machine are shown in Table 5.1. Next, we acquire the HAMs collected for each test set. Since each review is annotated by three people, we have three sets of HAMs: HAM_1, HAM_2, and HAM_3. Consensus among the three, CAM and SAM, are computed as per Defs. 2.4 and 2.5. Then we measure the Behavioral Similarity between human and machine. The amount of overlap in the selected words are presented in Table 5.2.

We observe that accuracy and similarity both decrease as the review-length increases and the classification task becomes more difficult for both humans and machine learning models. We identify two reasons for this: First, when a review is long, the prevailing opinion is usually not obvious at first glance and may require more intensive reading and contemplating. Second, the reviewers are more likely to state conflicting facts and opinion in long reviews. This, in turn, creates distracting and hard-to-read text. Compared to unidirectional model, bidirectional RNN with attention consistently rates closer to human attention. This is most striking for the Yelp-50 subset. This can be explained with the fact that bidirectional RNNs possess information from both directions of the text similar to humans.

For all three subsets, Yelp-50, Yelp-100, and Yelp-200, behavioral similarity for Consensus Attention Map is higher than all three HAMs. This is an important result because it indicates that the words all annotators agreed to be important are selected by machine attention too, whereas more subjective selections do not always get high attention from machine, indicated by lower SAM similarity.

Finally, we compare similarity of these three sets of HAMs. Even though human-to-human similarity is usually higher than human-to-machine similarity (as expected), the numbers still far from being close to 1. This confirms the subjectivity of human attention. Also, note that human-to-human similarity decreases as the review length increases.

We observe that the performance of the rationale-based models degrades more

sharply as the review-length increases. One conflicting interest with the rationale-based models is that the more words the model selects, the accuracy becomes higher. As our goal is to compare human attention with machine-generated attention for model interpretability, we optimize the model not only for accuracy but also for the number of selected rationales. We aim to generate roughly an equal number of words selected by both human annotators and machine-generated rationales. Hence, we force the rationale-models to pick fewer words by tuning the selection lambda accordingly. This gives a comparative advantage to attention-based models against rationale-based models, as the rationale model is a hard-attention mechanism. In addition, rationales are better suited for sentence-level tasks as they encourage consecutive selection as opposed to the behavior of attention.

5.6.3 Lexical Similarity Analysis

Next, we analyze if humans and neural networks pay more attention to words from particular lexical categories using Adjusted Lexical Similarity score.

Lexical Similarity results, presented in Table 5.3, are consistent with Behavioral Similarity in that bidirectional model with attention is most similar to human (0.91 for Yelp-50 and 0.84 for Yelp-100). Rationales model follows bidirectional RNN, and unidirectional RNN is the least similar model to human. Overall, lexical similarity to human decreases for all models, as the reviews become longer.

Next, we inspect which lexical categories are selected more heavily by human and machine. For this, we provide relative frequency of lexical categories for human-selected words, machine-selected words (bi-RNN), and overall relative frequency of this tag within the dataset. Adjectives (Human:0.24 bi-RNN:0.23 Overall:0.02), comparative adjectives (Human:0.002 bi-RNN:0.001 Overall:0.0001), and nouns (Human:0.38 bi-RNN:0.37 Overall:0.09) are among the lexical categories that humans and bi-RNN models favor heavily. Similarly, personal pronouns are rarely selected by neither humans nor bi-RNN models (Human:0.005 bi-RNN:0.005 Overall:0.01).

	Yelp-50		Yelp-100		Yelp-200	
	Lexical Sim.	Adjusted LS	Lexical Sim.	Adjusted LS	Lexical Sim.	Adjusted LS
Random Attention	0.85 ± 0.006	-	0.84 ± 0.013	-	0.90 ± 0.010	-
RNN Attention	0.93 ± 0.015	0.54	0.91 ± 0.007	0.44	0.93 ± 0.005	0.37
Bi-RNN Attention	0.99 ± 0.005	0.91	0.98 ± 0.013	0.84	0.93 ± 0.003	0.36
Rationales	0.95 ± 0.012	0.66	0.93 ± 0.027	0.53	0.90 ± 0.002	0.05

Table 5.3: Lexical Similarity and Adjusted Lexical Similarity of human attention to machine on varying review length. (Adjusted LS 0:no similarity, 1:perfect similarity)

	CSSR_p	CSSR_n
Human	0.06	0.20
RNN Attention	0.06	2.28
Bi-RNN Attention	0.04	0.19
Rationales	0.08	0.44

Table 5.4: Cross-sentiment Selection Rates for positive and negative reviews for Yelp-50 dataset.

5.6.4 Cross-sentiment Selection Rate Analysis

Finally, we compute CSSR scores, presented in Table 5.4, to evaluate the context-dependency of sentimental polarity for human and machine attentions. Our observations for Yelp-50 dataset are as follows. By human annotators, almost exclusively positive words are selected if the overall review sentiment is positive. For negative reviews, higher number of positive words are selected than negative words ($\text{CSSR}_p = 0.06, \text{CSSR}_n = 0.20$). Among the neural network models, the bidirectional RNN once more behaves most similar to human annotators with $\text{CSSR}_p = 0.04$ and $\text{CSSR}_n = 0.19$. RNN model's approach differs from that of human's and bi-RNN's. Even though the behaviour is similar for positive polarity ($\text{CSSR}_p = 0.06$), the opposite is true for negative polarity. In fact, positive words selected 2.28 times more than negative words in negative reviews, which is counter-intuitive. For the Rationales model, CSSR_p is 0.08 and CSSR_n is 0.44. This indicates that Rationales model is more similar to human attention than RNN model with attention. We observe similar trends for the Yelp-100 and Yelp-200 datasets.

Great place to go to have some drinks and food with a group of friends before heading out for the night. Service is mediocre and most of the menu is on the wall which was frustrating but the menu is very diverse. Really enjoyed the wagu carpaccio and the butterfish. Toro and uni were very fresh although the portions were small. One of the coolest things about this place is the fact that the pitchers of beer have a ice chamber in the center to keep your beer colder for longer! Genius! The wait can be long so come early.

Great place to go to have some drinks and food with a group of friends before heading out for the night. Service is mediocre and most of the menu is on the wall which was frustrating but the menu is very diverse. Really enjoyed the wagu carpaccio and the butterfish. Toro and uni were very fresh although the portions were small. One of the coolest things about this place is the fact that the pitchers of beer have a ice chamber in the center to keep your beer colder for longer! Genius! The wait can be long so come early.

Figure 5.3: Human attention is highly subjective. Some annotators tend to select only a few words, whereas others choose entire sentences.

5.6.5 Additional Analysis Results

An example visualization of the attention maps annotated by human annotators and machine learning models is provided in Figure 5.4. The agreement between human annotators and all machine learning models can be considered high in this example, as there are many mutual selections.

Another example is provided in Figure 5.3, demonstrating the attention maps provided by two different annotators for the same review. This is an extreme example of the subjectivity of human attention. The first annotator only highlights individual words with the strongest cues of sentiment, whereas the second annotator sometimes selects entire sentences when they indicate a sentiment.

Table 5.5 shows the distribution of selected words over lexical categories for Human (CAM), Machine (bi-RNN), and the entire corpus for the Yelp-50 subset. Any divergence in the Human and Machine columns from the Corpus column indicates a tendency of selection for a lexical category. For example, adjectives are selected very heavily by both Human and Machine, even though they only make 0.02 of all words in the dataset.

Lexical Category	Human	Machine(bi-RNN)	Corpus
Coordinating conjunction	0.0000	0.0098	0.0147
Cardinal number	0.0098	0.0077	0.0043
Determiner	0.0112	0.0168	0.0312
Existential there	0.0000	0.0000	0.0000
Foreign word	0.0000	0.0000	0.0000
Preposition or subordinating conjunction	0.0266	0.0084	0.0298
Adjective	0.2374	0.2269	0.0201
Adjective, comparative	0.0021	0.0014	0.0002
Adjective, superlative	0.0252	0.0287	0.0016
List item marker	0.0000	0.0000	0.0000
Modal	0.0035	0.0000	0.0030
Noun, singular or mass	0.3838	0.3711	0.0950
Noun, plural	0.0000	0.0000	0.0000
Proper noun, singular	0.0000	0.0000	0.0000
Proper noun, plural	0.0413	0.0665	0.0154
Predeterminer	0.0000	0.0000	0.0000
Possessive ending	0.0000	0.0000	0.0000
Personal pronoun	0.0056	0.0049	0.0141
Possessive pronoun	0.0035	0.0028	0.0067
Adverb	0.1296	0.0931	0.0277
Adverb, comparative	0.0070	0.0000	0.0014
Adverb, superlative	0.0000	0.0000	0.0000
Particle	0.0000	0.0000	0.0000
Symbol	0.0000	0.0000	0.0000
to	0.0035	0.0007	0.0077
Interjection	0.0000	0.0000	0.0000
Verb, base form	0.0196	0.0028	0.0098
Verb, past tense	0.0070	0.0609	0.0148
Verb, gerund or present participle	0.0357	0.0462	0.0053
Verb, past participle	0.0455	0.0455	0.0083
Verb, non-3rd person singular present	0.0000	0.0028	0.0023
Verb, 3rd person singular present	0.0007	0.0021	0.0065
Wh-determiner	0.0000	0.0000	0.0005
Wh-pronoun	0.0007	0.0000	0.0005
Possessive wh-pronoun	0.0000	0.0000	0.0000
Wh-adverb	0.0007	0.0007	0.0012

Table 5.5: Distribution over lexical categories for human-selected words, machine-selected words, and the entire corpus.

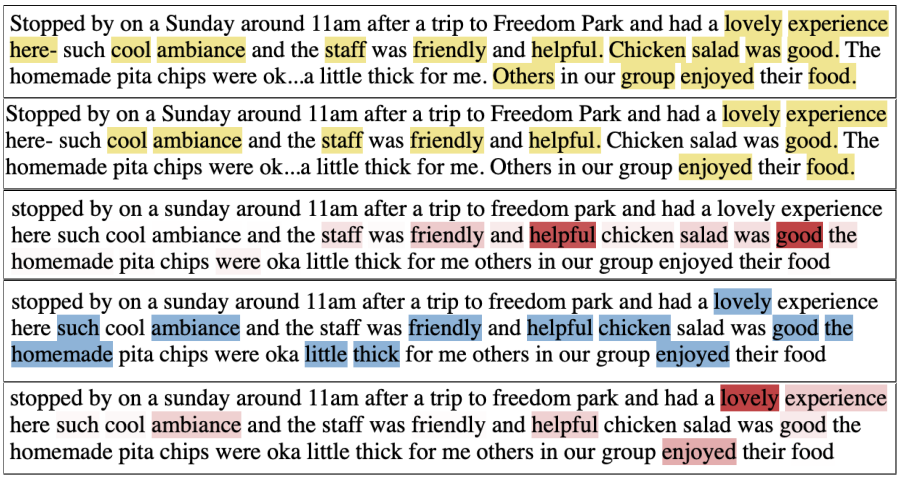


Figure 5.4: Visualizations of attention maps by human annotators and machine learning models. From top to bottom: first human annotator, second human annotator, RNN, bi-RNN, Rationales.

5.7 Discussion

Recent papers, including our work, take strides at answering the question if attention is interpretable. This is complicated by the fact that "interpretability" remains a not well-defined concept.

Attention adds transparency. [58] defines *transparency* as overall human-understanding of a model, i.e., why a model makes its decisions. Under this definition, attention scores can be seen as partial transparency. That is, they provide a look into the inner workings of a model, in that they produce an easily-understandable weighting of hidden states [108].

Attention is not faithful. [87] defines *explainability* as a plausible, but not necessarily faithful, reconstruction of the decision-making process. Whether adversarial attention scores exist that result in the same predictions as the original attention scores helps us understand if attention is *faithful*. With their empirical analyses, [92] and [41] show that attention is not faithful.

Rationale models for human-like explanations. [85] argues that *explanations* are post-hoc descriptions of how a system came to a given conclusion. This raises the question of what makes a good explanation of the behavior of a machine learning system. One line of research offers these explanations in the form of binary *rationales*, namely, explanations that plausibly justify a model's actions [7, 57].

Our approach at attention as human-like explanations. In claiming *attention is explanation*, it is seen to mimic humans in rationalizing past actions. In our work, we approach interpretability from this human-centric perspective. We develop a systematic approach to either support or refute the hypothesis that attention corresponds to human-like explanations for model behavior. Based on our comparative analyses, we provide initial answers to this important question by finding insights into the similarities and dissimilarities of attention-based architectures to human attention.

Towards additional tasks beyond text classification. Confidently concluding whether attention mimics human requires tremendous efforts from many researchers with human data to be collected via a well-designed data collection methodology, both labor-intensive and costly task. In this work, we thus focus on one task, namely, sentiment classification, and collect HAM for this task and on a single dataset. We invite other researchers to continue this line of research by exploring other tasks (e.g., question answering).

5.8 Summary

For this task, we collect human attention maps for text classification through crowd-sourcing on the Yelp Dataset. This new resource will be made available to the research community. We use our collected human attention dataset for quantifying the similarities between human and attention-based neural network models. We take into account different linguistic properties while designing the similarity metrics. Results indicate significant similarities between bidirectional RNNs with human attention with regard to overlap in the word selections, selecting words from the same lexical categories and, selecting contrastive words to the chosen sentiment. Our findings open promising future research opportunities ranging from supervised attention to the design of human-centric attention-based explanations.

This task is resulted in the following publication:

- C. Sen, T. Hartvigsen, B. Yin, X. Kong, E. Rundensteiner, "Human Attention Maps for Text Classification: Do Humans and Neural Networks Focus on the Same Words?", n Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 4596-4608. 2020.

Chapter 6

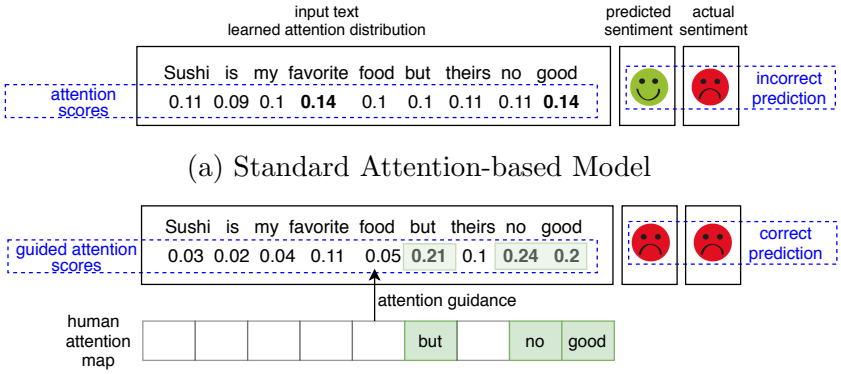
Human-guided Attention

6.1 Motivation

Attention mechanisms, combined with deep sequence models, are the architecture of choice for a vast array of text classification tasks [91, 114]. Typically, attention mechanisms are incorporated as auxiliary *unsupervised* sub-problems of classification. For example, when classifying documents, the attention functions generate one attention score per word in the document. However, the attention scores themselves are only predicted with respect to the overall class predictions – instead of being explicitly guided at the granularity of individual attention values. This approach thus only serves to improve the ultimate classification accuracy, while disregarding the prevalent requirement in many domains of providing *explainability* of the model decisions.

Interpretable methods for text classification are essential in many settings. For example, doctors must know which words an algorithm for clinical note-driven diagnosis utilizes in order to trust the prediction. However, the prevalent assumption that unsupervised attention mechanisms inherently create explainable models has recently been debunked [41, 92]. Researchers have begun to demonstrate that, instead, attention maps that look like they were generated by humans are actually interpretable [7, 91]. Thus, we must aim to *supervise attention mechanisms to encourage such human-likeness* to achieve the important explainability requirement.

Recent works have begun explicitly learning attention functions by *supervising* the attention mechanism itself [13, 63, 70, 117]. These methods rely on hand-picked



(a) Standard Attention-based Model

(b) HUG-Based Model

Figure 6.1: In the traditional attention-based model (top), attention learning is unguided. Our proposed human-guided attention model HUG (bottom) learns attention scores based on explicit attention-granularity human feedback. The guided model is more interpretable as it is capable of picking up words that humans find more important wrt end-task. It is also more accurate.

lists of target words that should receive the most attention. These "interest words" are selected using domain knowledge [60]. Thereafter, the learning methods penalize attention that deviates from these lists [70, 117]. This strategy only guides the attention indirectly, since the same list of interest words are applied to all text. Thus, this approach is rigid and does not lead to the 'human-like' explanations that are necessary to render classifiers interpretable [7]. Attention mechanisms should instead be data-driven and vary their attention depending on the input.

Next we introduce the notion of human-guided attention. Given a corpus of training text, ground truth class label for the classification of each text, and token-level labels for human attention, the goal is to learn a model that solves the primary text classification task accurately while *also* optimizing the attention function to become similar to human attention. During *inference*, the trained model must then assign an attention weight to each word of the input text with respect to classification target *without* human guidance. As such, the learned attention scores serve as explanation regarding what words are critical for reaching the final classification decision.

Figure 6.1 depicts the human-guided attention model and contrasts it to a standard attention architecture. We postulate that classification and human-like attention on the *same* task should be complementary. Thus, in addition to bringing

human-like reasoning into sequential deep network models, we study the hypothesis that human-guided attention, jointly learned with a classification task, improves the performance of the primary classification task as well.

Designing human-guided attention is challenging for the following reasons.

- *Limited availability of token-level labels.* Token-level human attention labels are required to guide attention mechanisms. Since collecting such data can be costly and time-consuming, a solution must be effective even when given only a small amount of guidance.

- *Trade-off between model accuracy and explainability.* While interpretable models may provide accurate and authentic explanations, the question of when/if this comes at the cost of sacrificing prediction performance remains open [26]. Thus, designing a methodology that makes attention more interpretable while continuing to train for effective prediction accuracy is vital.

- *Model interpretability is not well-defined.* Devising *interpretable* deep models is vital for many domains, healthcare being one eminent example [16, 93]. However, achieving and measuring the interpretability can be complex, since interpretability itself is not well-defined. Thus, it is essential that we establish metrics capable of quantifying the *interpretability* of a model.

We propose a Human-Guided attention mechanism, or HUG, that learns to combine machine-inferred attention with human guidance to achieve improved classification accuracy while also offering human-like explanations for the model's behavior. The HUG mechanism can be paired with any sequence-representation learning architecture (e.g.RNN, LSTM, GRU, BERT) to model text with respect to a classification task. HUG-augmented deep models incentivize the learned attention function to remain similar to human attention annotations, resulting in more human-like attention outcomes. We jointly optimize the dual supervision objectives at both the word and document granularity to learn better the overall language representations and the corresponding attention functions. To quantify the interpretability of HUG-based models, we define a *human-likeness* metric. We then evaluate the model performance based not only on the text classification accuracy but also on the degree to which the explanations made by the model are

human-like. We evaluate our HUG architecture by pairing it with core sequential deep learning models, including LSTM, GRU, and BERT.

Our experiments using existing human attention map datasets [1, 91] confirm that adding human-like attention supervision is a win-win in that it not only yields more human-like attention (i.e.inferred attention more accurately matching that of humans) but also boosts classification performance. This points to a promising direction for future research in this new area of human-like attention mechanisms.

Our contributions are as follows:

- We propose the first data-driven human-guided attention mechanism, HUG, and a deep learning architecture that learns important words for a given classification decision, as declared by humans, to serve as explanations for model decisions.
- We demonstrate that explanations provided by HUG-based models are up to 28% more human-like, with the added benefit that the classifications are up to 5.8% more accurate than state-of-the-art unguided attention models.
- We show that attention supervision can be used to regularize attention functions. Using our approach, a small training dataset can achieve equal accuracy to unguided models trained using over 40% more data.

6.2 HUG Framework

In this section, we first formally define the attention-guidance problem, followed by describing our proposed method HUG framework.

6.2.1 Notation and Problem Definition

Given a set of N documents $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_N\}$, each document consisting of T words $\mathcal{D}_i = [w_{i1}, \ldots, w_{iT}]$ and a set of class labels for the document set $y = \{y_1, \ldots, y_N\}$ where $y_i \in \{0, 1\}$ indicating the correct document class, the *sequence classification* task is to parameterize a function $f(\theta)$ that maps $\mathcal{D}_i \rightarrow y_i$ for instances on which it was not trained.

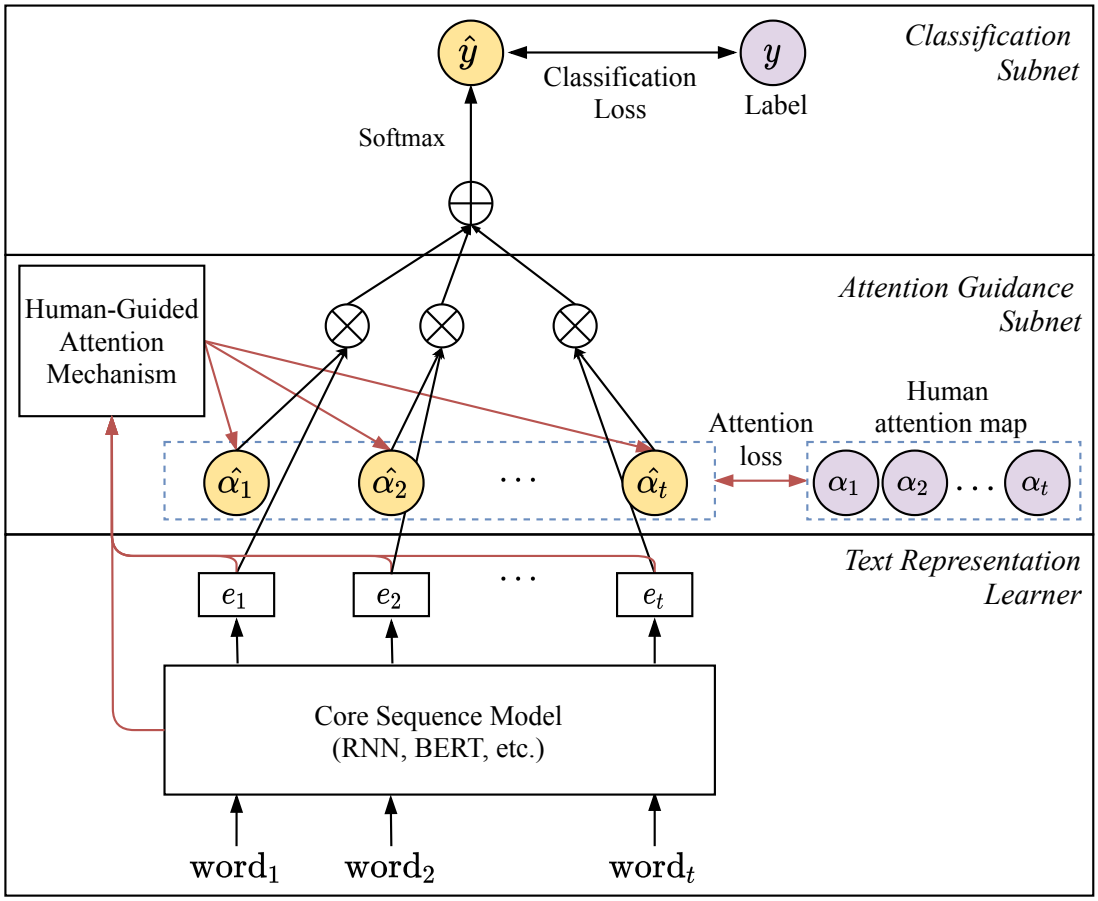


Figure 6.2: Overall architecture of the HUG Framework

An attention mechanism computes a probability distribution to serve as "importance weights" for each word in a document. These weights are then used to compute a weighted average of a vector representation of each word. When the input is a document, an *Attention Map* $\text{AM} = [\alpha_1, \ldots, \alpha_T]$, computed by an attention function, is a vector of length T with each entry α_t being associated with a word in the corresponding position of the associated text. The value of α_t indicates the amount of attention the corresponding word w_t receives with respect to a classification task.

A *Human Attention Map* $\text{HAM} = [\alpha_1, \ldots, \alpha_T]$ is a binary attention map produced by a human, where each entry with a set-bit $\alpha_t = 1$ indicates that the corresponding word receives high attention (while 0 means no or low attention). A *Machine Attention Map* $\text{MAM} = [\hat{\alpha}_1, \ldots, \hat{\alpha}_T]$ is an attention map *predicted* by a neural network model. If computed through soft-attention, a MAM corresponds to an AM of continuous values that capture a probability distribution over the words.

In our problem setting, we assume $f(\theta)$ includes an attention mechanism and we add constraints during estimation of $f(\theta)$ to encourage the learned attention map to become similar to human attention. In this new setting, in conjunction with a document set $\mathcal{D}$, for each document $\mathcal{D}_i$, we are given a label tuple $Y_i = (y_i, \text{HAM}_i)$, where $y_i \in \{0, 1\}$ corresponds to the class label of the document $\mathcal{D}_i$ and $\text{HAM}_i = [\alpha_{i1}, \ldots, \alpha_{iT}]$ to the human attention map of this document $\mathcal{D}_i$ where $\alpha_{it} \in \{0, 1\}$, with 1 indicates that the corresponding word received high attention.

Let us assume that a parameterized function $f(\theta)$ that learns how to map $\mathcal{D}_i$ to a class label y_i while also concurrently learning a set of attention scores $\text{MAM}_i = [\hat{\alpha}_{i1}, \ldots, \hat{\alpha}_{iT}]$. Our task then is to jointly learn the function $f(\theta)$ while minimizing the difference between HAM_i and MAM_i for all documents $\mathcal{D}_i$ where $i = 1, \ldots, N$. For readability, we henceforth describe our method for a single document and thus drop the subscript i whenever it is unambiguous. Table 6.1 summarizes the notation used throughout the paper.

6.2.2 Overview of the Proposed HUG Framework

Our proposed HUG (Human-Guided) attention framework consists of three subnetworks, as shown in Figure 6.2. The bottommost is the *Core Sequence Model*. This block can be realized with any sequential deep learning architecture such as

Table 6.1: Basic Notation

Notation	Explanation
N	Total number of documents
T	Total number of words in a document
D_i	i-th document
w_{it}	word t in document i
x_{it}	vector representation of word t in document i
e_{it}	transformed vector of word t in document i
Y_i	Label tuple for D_i
y_i	Class label for D_i
$\hat{y}_i$	Predicted class for D_i
HAM_i	Human attention labels for all words in D_i
MAM_i	Machine-generated attention scores for all words in D_i
α_{it}	True attention score for word t in document i
$\hat{\alpha}_{it}$	Predicted attention score for word t in document i
	where $i = \{1, \ldots, N\}$ and $T = \{1, \ldots, T\}$.
	Subscript i is dropped throughout the paper for simplicity.

an RNN or a Transformer. The purpose of this layer is to encode input sequences into learned representations. This layer is followed by an attention mechanism to generate a Machine Attention Map (MAM) for each input token. The resulting MAM structure is utilized by the following two subnetworks: The *Classification Subnet* and the *Attention Guidance Subnet*. The Classification Subnet employs the MAM to compute a context vector that will be further used to generate the probability of the document $\mathcal{D}$ belonging to class c. Using the correct class label y, this subnet then computes the sequence classification loss. The Attention Guidance subnet models the distance between the MAM and the corresponding HAM. This distance is then minimized jointly with the sequence classification objective. Each of these steps is described in the following sections.

6.2.3 Core Sequence Model

We implement the following algorithms as the core sequence model.

HUG-RNN. One common and powerful architecture for document classification is an RNN combined with an attention mechanism [5, 114]. Following this architecture, the HUG-RNN model first utilizes an encoding layer to map words into real-valued vector representations where semantically-similar words are mapped close to one another. We use a pre-trained word embedding set ϕ for this mapping: $x_{it} = \phi w_{it}$. HUG-RNN then employs a recurrent layer to embed vector representations of words into hidden states, processing words one at a time. In our experiments, we use both LSTM and GRU memory cells.

Assuming that Γ is the recurrence function (e.g.LSTM or GRU) and x_t is the embedded t-th word from the document $\mathcal{D}$, HUG-RNN is modeled as:

$$e_t = \Gamma(x_t, e_{t-1}) \tag{6.1}$$

$$u_t = \Phi(We_t + b) \tag{6.2}$$

$$\hat{\alpha}_t = \frac{\exp(u_t^\top u)}{\sum_t \exp(u_t^\top u)} \tag{6.3}$$

where W, b, and u are trainable parameters, Φ is the hyperbolic tangent function, and $\hat{\alpha}_t$ are the attention scores for each word, computed through a softmax function as shown in Equation 6.3. MAM $= [\hat{\alpha}_1, \ldots, \hat{\alpha}_T]$ is then utilized by the further layers of HUG-RNN.

HUG-BERT. HUG-BERT first employs a transformer-encoder architecture[105] to encode words, initialized with a pre-trained BERT model[23]. This layer is followed by an attention mechanism to output a Machine Attention Map (MAM) for each document. Following the standard practice in BERT-based architectures, the first token of the input is the special token '[CLS]'. '[SEP]' token is added to the end of the input sequence to denote the end. '[PAD]' token is used to pad the sequence in case the input sequence is shorter than the maximum input length supported by the BERT model. HUG-BERT generates two outputs. First is a sequence of learned word representations $[e_1, ...e_T]$ for each input token. Second is a vector representation r for the whole input document. This vector r corresponds to the output of the '[CLS]' token further processed by a linear layer and a tanh activation function.

$$[e_1, ...e_T], r = \text{BERT}([w_1, ...w_T]) \tag{6.4}$$

Then the attention score of each token is defined as the similarity of e_t with the vector representation r. We compute the normalized attention scores $\hat{\alpha}_t$ through a Softmax function as follows:

$$\hat{\alpha}_t = \frac{\exp\left(e_t^\top r\right)}{\sum_t \exp\left(e_t^\top r\right)} \tag{6.5}$$

$\text{MAM} = [\hat{\alpha}_1, \ldots, \hat{\alpha}_T]$ is then utilized by the further layers of the HUG framework.

6.2.4 Classification Subnet

Using the Machine Attention Map (MAM) and the learned representations of words e generated by the core sequential model, the Classification Subnet first computes a context vector c as follows:

$$c = \sum_t \hat{\alpha}_t e_t \tag{6.6}$$

The context vector c models a dense embedding for the document. The Classification Subnet uses c and assigns a probability to each possible class. We use the cross-entropy loss as the sequence classification objective function where $\hat{y}$ is the prediction and y is the ground truth label, as shown in Equation 6.7.

$$J_c(y, \hat{y}) = -(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})) \tag{6.7}$$

6.2.5 Attention Guidance Subnet

The main goal of the Attention Guidance Subnet is to optimize the learned attention scores to be as close as possible to human attention. This way, attention scores can be interpreted as human-like reasonings for the final classification decision, adding HUG-based models explainability.

To this end, we take the squared error as the general loss of the attention at the word level to supervise the learning process.

$$J_a(\text{HAM}, \text{MAM}) = \sum_{t=1}^{T} (\hat{\alpha}_t - \alpha_t)^2. \tag{6.8}$$

This objective optimizes the model to assign correct importance scores to every word. By providing word-level supervision to the document classification model, we are able to teach it to focus on the most relevant areas selected by humans and thereby improve the quality of document representations along with overall performance.

It is worth noting that, special tokens, such as '[CLS]', '[SEP]', and '[PAD]' are invisible to human annotators (if core sequence model is BERT). Thus, their corresponding human attention weights are always zero. Also, the tokenizer used by the BERT model is WordPiece[110], which sometimes split a word into several tokens. These generated tokens are assigned with the same human attention score as the original word.

6.2.6 Joint Training of the HUG framework

In the HUG framework, Classification Subnet and Attention Guidance Subnet are jointly trained. Thus, we define a joint loss function in the training process upon the losses specified for different subnets as follows:

$$J(\theta) = \sum (J_c(\hat{y}, y) + \lambda J_a(\text{HAM}, \text{MAM})) \tag{6.9}$$

where θ denotes, as a whole, the parameters used in our model, and λ is the hyper-parameter for striking a balance among the sequence classification and attention guidance. By integrating these two objectives similar to the multi-task setting, we let the two tasks aid each other's training.

While the sequence classification is the primary task in HUG architecture, in some domains such as healthcare, explainable models are equally important. The hyper-parameter λ is a tunable parameter, and it can be optimized to put more emphasis either on the primary task accuracy or model explainability. Recent research has shown that any errors encoded in the attention function may propagate to classification decisions [3], thus impairing model performance. Hence, we speculate that forcing attention to become more similar to human attention should benefit the primary task accuracy as well.

During the training, the HUG framework requires HAMs to learn a human-like attention function. However, in the inference time, it does not require attention labels.

6.2.7 Attention Supervision as a Regularizer

Obtaining HAMs for the entire training corpus can be costly. The HUG framework does not necessarily require the whole dataset to be accompanied by HAMs. Instead, a small portion of the data where HAMs exist and a large part of the data with no accompanying HAMs can be used together to train the model as follows.

At every epoch, we sample a batch of training examples either from the set where HAMs are available or from the set where only documents exist with no human annotations as determined by a coin flip, the bias of which is determined by the data availability. If the batch is sampled from the former set, the sequence classification loss is computed using Equation 6.7. Otherwise, the joint loss is computed through Equation 6.9. We provide the pseudo-code for this learning paradigm in Algorithm 1.

6.3 Experiments

To evaluate our proposed framework, we use publicly available text datasets containing human annotations at the word level.

Algorithm 1: How to use HUG framework for regularizing the attention function if limited HAM availability.

```
dataset_d ← Document set with HAMs;
dataset_l ← Document set without human annotations;
batch_d ← generate batch pool from dataset_d;
batch_l ← generate batch pool from dataset_l;
batch_pool ← batch_d ∪ batch_l;
for each epoch in max_epoch do
    for each batch in batch_pool do
        if batch ∈ batch_d then
            Compute Loss using Equation 6.7;
        else
            Compute Loss using Equation 6.9;
        end
        Optimize Loss;
    end
end
```

6.3.1 Sentiment Classification Task on YELP-HAT Dataset

Yelp Human Attention Dataset. YELP-HAT provides human attention maps for a subset of reviews from the Yelp dataset [91]. These human attention maps in YELP-HAT are collected for the classification task of rating a review as positive or negative on Amazon Mechanical Turk. Participants are asked to highlight the words that are indicative of the review's overall sentiment. For each review in the YELP-HAT dataset, three annotations are collected from different subjects.

Sentiment Classification Task. YELP-HAT dataset also contains a binary sentiment label for each review. Using these as class labels, we focus on predicting the sentiment as either positive or negative. The dataset is pre-processed to remove punctuation and lowercased. To encode words into vectors, we use pre-trained Glove embeddings with 100-dimensions [73]. The dataset contains 1000 reviews, with the length varying between 50-75 words. 30% of this dataset is used for reporting all evaluation metrics. Class distribution is balanced.

Extracting Human Attention Maps. Human attention is highly subjective. Each person can have a unique approach to which words are indicative of a sentiment. For obtaining a reliable representation of human attention, we use the

following operations proposed by [91] to coalesce HAMs by multiple annotators into aggregated HAMs. **Consensus Attention Map (CAM)** is computed through a bitwise AND operation of the HAMs. It contains words that all three annotators agreed on being important. **Super Attention Map (SAM)** is computed by a bitwise OR operation of the HAMs. It includes every word highlighted by at least each annotator. We postulate that CAMs are a more reliable source of human reasoning for attention guidance as they contain high-confidence words. We also experiment with SAMs because even if a single annotator considers a word relevant to the sentiment decision, that word is likely to rank higher in the importance than the words that none of the annotators selected. On the other hand, SAMs may introduce noise into the attention signals as not all annotators provide reviews of equal quality.

6.3.2 Heart Disease Prediction Task on N2C2 Dataset

N2C2 Dataset. N2C2 NLP Research Data Sets contain unstructured notes from the Research Patient Data Repository at Partners Healthcare[1]. From this clinical note repository, we use the 2014 challenge data, consisting of a set of medical documents that track the progression of heart disease in diabetic patients. Each clinical note in this dataset is annotated by expert annotators to indicate the presence and progression of a disease (diabetes or heart disease), associated risk factors, and the time they were present in the patient's medical history. Annotations are from a single annotator per note.

Hearth Disease Prediction Task. We focus on predicting heart disease. For each patient in the dataset, if there is a clinical note with a heart disease annotation (indicated by the CAD tag in the dataset), we assign all notes belonging to this patient to the positive class. Patients with no heart disease mention are assigned to the negative class. Then we train a model that inputs every individual clinical note and predicts whether this note belongs to a heart-disease patient. For mapping words into vectors, we use BioMed embeddings [78]. N2C2 dataset contains 520 clinical notes in its training split and 511 clinical notes for testing, assigning notes from the same patient into either the training or test set. This test set is used for

[1] https://n2c2.dbmi.hms.harvard.edu

reporting all evaluation metrics. Class distribution is balanced.

Extracting Human Attention Maps. We use all heart disease-related annotations to create human attention maps. These annotations include remarks of patients having heart disease (e.g., "coronary artery disease") or indirect mentions (e.g., "unstable angina," "PLAVIX" - a blood thinner used to prevent heart attack).

6.3.3 Compared Methods

We compare the following model performances:

- **Vanilla model**. This model refers to the architecture with no attention guidance. We experiment with three core sequence models: LSTM, GRU, and BERT. Vanilla models for LSTM and GRU correspond to each network paired with an unguided attention mechanism. Vanilla BERT model corresponds to a standard BERT architecture as it is used in the literature with only native BERT attention [23].

- **Pre-fixed guidance**. A common approach for supervising the attention mechanism is to use a hand-picked list of words to receive high attention[63, 70, 117]. For each dataset, using the domain knowledge we have, we construct a list of words, and we use these as attention guidance signals.

- **HUG-X**. This model represents the proposed architecture HUG paired with one of the core sequence models X. For example, HUG-BERT represents our HUG mechanism paired with the BERT model.

6.3.4 Metrics

We use two metrics to compare model performances.

Accuracy. We use accuracy for measuring the sequence classification performance. All datasets we use are balanced. Thus the lower bound is 50%.

Human-Likeness Score. For evaluating the attention guidance performance and interpretability of models, we design a metric, called *human-likeness score*. For two attention mechanisms (i.e.human and machine-learned attention) to be

Table 6.2: Performance comparison of baseline methods and the proposed method for sentiment classification task.

Core Sequence Model	Compared Methods	Metrics		
		Accuracy	Human-likeness	Combined Performance
LSTM Models	Unguided Attention	83.6 ± 0.61	49 ± 1.5	66.55
	Prefixed Guidance	86.1 ± 0.58	60 ± 2.1	73.05
	HUG-LSTM$_{\text{SAM}}$	88.5 ± 0.41	69 ± 1.8	78.75
	HUG-LSTM$_{\text{CAM}}$	$\mathbf{89.4} \pm 0.69$	$\mathbf{76} \pm 0.9$	**82.70**
GRU Models	Unguided Attention	86.3 ± 0.43	52 ± 2.1	69.15
	Prefixed Guidance	88.2 ± 0.51	62 ± 2.3	75.10
	HUG-GRU$_{\text{SAM}}$	89.3 ± 0.43	69 ± 1.8	79.15
	HUG-GRU$_{\text{CAM}}$	$\mathbf{90.1} \pm 0.67$	$\mathbf{78} \pm 1.5$	**84.05**
BERT Models	Unguided Attention	95.0 ± 0.33	–	–
	Prefixed Guidance	93.4 ± 0.38	61 ± 7.1	77.20
	HUG-BERT$_{\text{SAM}}$	95.3 ± 0.41	84 ± 5.7	89.65
	HUG-BERT$_{\text{CAM}}$	$\mathbf{95.6} \pm 0.36$	$\mathbf{89} \pm 1.5$	**92.30**

similar, they must put attention on the same parts of the text. Thus, we quantify the overlap in words selected by human annotators and by the deep learning model.

Given a collection of attention maps HAM and MAM for a text dataset $\mathcal{D}$, the human-likeness score of the learned machine attention corresponds to the average pair-wise similarity between each $(\text{HAM}_i, \text{MAM}_i)$ vector pair $\forall i \in \mathcal{D}$ as defined below:

$$\text{PairwiseSim}_i = \text{AUC}(\text{HAM}_i, \text{MAM}_i)$$

$$\text{Human-likeness}(M, H) = \frac{1}{|\mathcal{D}|} \sum_i (\text{PairwiseSim}_i)$$

where $|\mathcal{D}|$ is the number of documents in the dataset $\mathcal{D}$. Intuitively, this corresponds to adopting the human attention vector as binary ground truth. That is, it measures how similar the machine-generated continuous vector is to this ground truth. AUC is between 0 and 1 with .5 representing no similarity, and 1 the perfect similarity.

Table 6.3: Performance comparison of baseline methods and the proposed method for Heart Disease Prediction Task

Core Sequence Model	Compared Methods	Metrics		
		Accuracy	Human-likeness	Combined Performance
LSTM Models	Unguided Attention	73.90 ± 1.21	41 ± 7.8	62.45
	Prefixed Guidance	72.27 ± 1.15	61 ± 0.5	66.64
	HUG-LSTM	$\mathbf{76.05} \pm 0.37$	$\mathbf{80} \pm 1.6$	**78.02**
GRU Models	Unguided Attention	72.73 ± 0.23	56 ± 3.1	63.39
	Prefixed Guidance	74.05 ± 0.69	69 ± 3.4	71.53
	HUG-GRU	$\mathbf{75.34} \pm 0.15$	$\mathbf{84} \pm 0.9$	**79.66**
BERT Models	Unguided Attention	78.16 ± 1.4	–	–
	Prefixed Guidance	78.08 ± 0.6	73 ± 7.5	75.54
	HUG-BERT	$\mathbf{78.47} \pm 1.5$	$\mathbf{85} \pm 4.1$	**81.74**

6.3.5 Experimental Results

6.3.5.1 Performance of HUG-based models

We first evaluate HUG-based models for a setting where word-level human annotation exists for the entire dataset, including the training and test sets. Even though the HUG architecture does not require HAMs during inference, with the availability of HAMs for both the training and test sets, we can quantify our model's interpretability. For the YELP-HAT dataset, we train two versions of the model: HUG-X_{CAM} utilizes Consensus Attention Maps as the ground truth for human guidance, whereas HUG-X_{SAM} utilizes Super Attention Maps. For the N2C2 dataset, only one HAM exists for each data instance. Accuracy and human-likeness score for these experiments are presented in Tables 6.2 and 6.3. These results show that all human-guided models achieve improved sequence classification accuracy and human-likeness compared to the baseline models.

For the sentiment classification task, HUG-$LSTM_{CAM}$ achieves the most substantial improvement in accuracy by 5.8%. While the proposed HUG-based models show improvement in the classification accuracy for all three core sequence algorithms, HUG-$BERT_{CAM}$ achieves the least increase. This is likely because the HUG-BERT model is pre-trained on large text corpus, whereas HUG-LSTM and HUG-GRU models are being trained from scratch on a small dataset. This causes the baseline BERT model to achieve an already high accuracy, which is challenging to improve on. As per human-likeness, all HUG-based models achieve significant gains (up to

28%) compared to the baseline models.

Using SAM vs. CAM for attention guidance leads to varying amounts of performance gain. While still accomplishing improved performance compared to baseline models, HUG_{SAM} models perform worse than HUG_{CAM} models. We observe this same trend for all three core sequence algorithms. This may be because SAMs include a high percentage of words selected as important in every document. As a result, they contain too little information for the model to learn which words indeed matter. CAMs, on the other hand, only include the words all three annotators agreed to be important. This filtered more confident human-intuition leads to the best results when used for attention guidance.

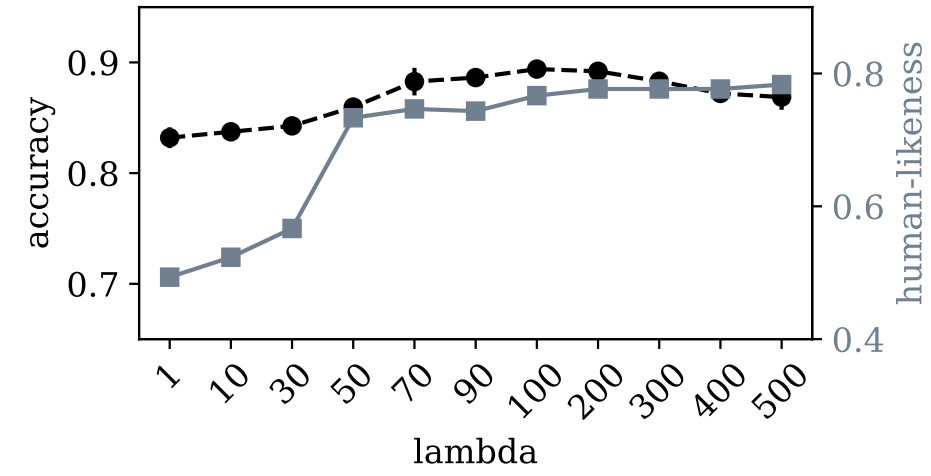
For the heart disease prediction task, we observe similar trends as for the sentiment classification task. We observe more significant gains in the classification accuracy for HUG-LSTM and HUG-GRU models compared to the HUG-BERT over the baseline methods. Improvement in human-likeness is extensive for all core algorithms.

6.3.5.2 Hyper-parameters Analysis

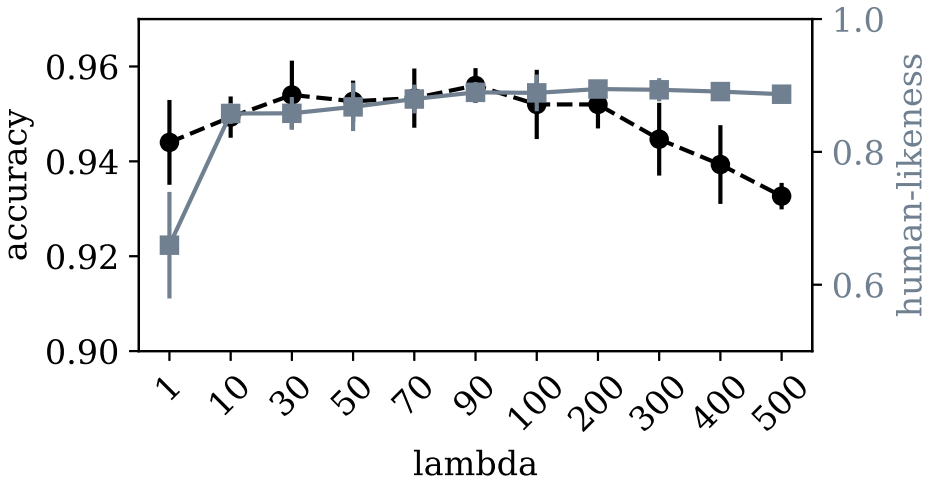
HUG model has one hyper-parameter, λ, which determines how much weight to put on the sequence classification versus the attention guidance. Next, we investigate how changing λ affects classification performance and human-likeness score.

Figure 6.3 shows experimental results for varying values of λ for HUG-LSTM and HUG-BERT models on the YELP-HAT dataset. We observe that if we keep putting more weight on the attention correctness, both accuracy and human-likeness steadily improve up to a point. Then, the human-likeness score keeps improving at the cost of accuracy. Optimum λ can be selected from the $[100, 200]$ range for HUG-LSTM and $[30 - 90]$ range for HUG-BERT as shown in Figure 6.3, since those λ values optimize both accuracy and human-likeness score.

The optimal value of λ can be decided depending on one's objective. One can prefer putting more importance on the accuracy or human-likeness score depending on the objective. For some domains, such as healthcare, even sacrificing some of the predictive power to gain more interpretability may be preferable. In such cases, λ can be tuned to highlight human-likeness even more heavily.



(a) HUG-LSTM



(b) HUG-BERT

Figure 6.3: Effect of λ on accuracy and human-likeness. Values from the [100,200] range for HUG-LSTM and [30-90] range for HUG-BERT maximize both accuracy and human-likeness. However, λ can be optimized to emphasize either accuracy or human-likeness much more heavily depending on the domain and objective.

Table 6.4: We assign a fixed budget for data collection and split this budget for collecting classification labels (cost=x) and word-level attention labels (cost=y). These two types of labels are assumed to have equal cost and the budget function is $x + y = 1$.

	Training Reviews (x)	Accompanying HAM (y)	Metrics		
			Accuracy	Human-likeness	Combined Performance
Baseline	100%	0%	0.834 ± 0.006	0.497 ± 0.004	0.665
Budget function: x+y=1	90%	10%	0.824 ± 0.002	0.512 ± 0.002	0.667
	80%	20%	0.834 ± 0.006	0.526 ± 0.012	0.680
	70%	30%	0.831 ± 0.004	0.58 ± 0.020	0.705

6.3.5.3 Attention Regularization

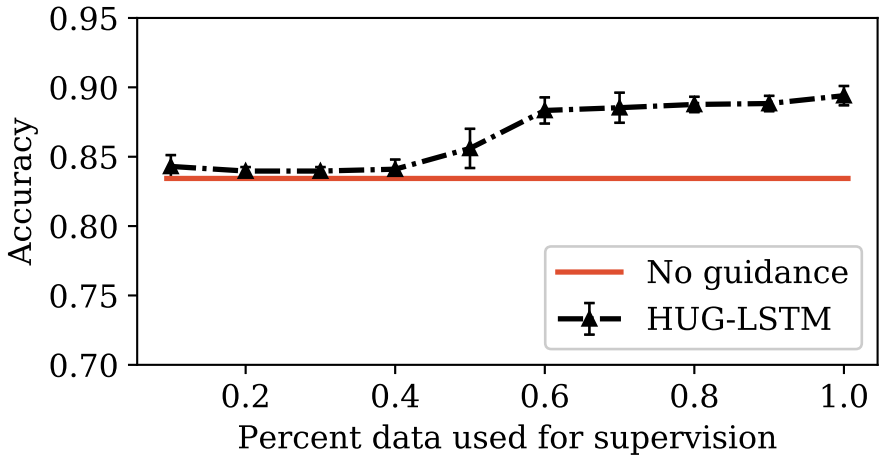
In Section 6.2.7, we describe how to use the HUG framework to regularize the attention function with human-guidance, when there is only limited data available for attention guidance. In this section, we investigate the performance of HUG models as an attention regularizer.

In these experiments, we focus on the sentiment classification task and HUG-LSTM model as there is a more significant gap between the baseline models and HUG-LSTM performance concerning classification accuracy. We use the same training set as in the previous experiments. However, we do not employ HAMs for the entire dataset. Instead, we change the percentage of *instances with HAMs* to be 10% to 90% of the total training data. Then, following the training procedure described in Algorithm 1, we train models and measure the performance of HUG-LSTM. We use CAMs for human guidance.

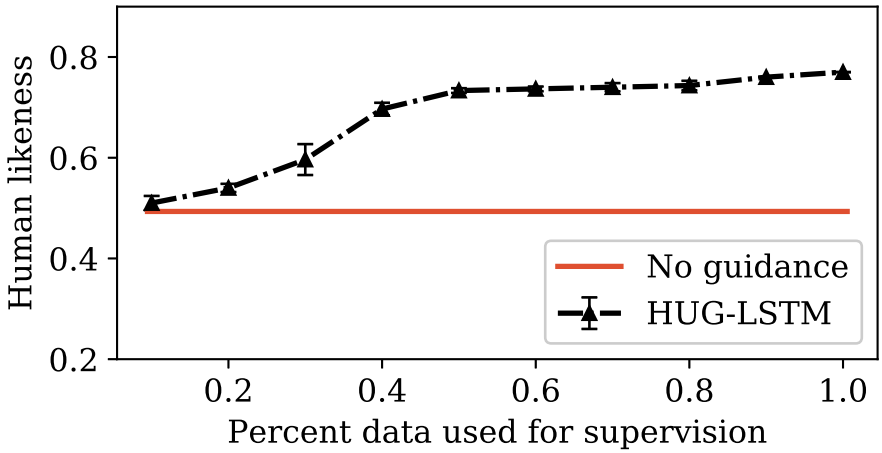
Results are presented in Figure 6.4. A significant conclusion is that having as little as 20% of the training data accompanied by HAMs leads to a 6% improvement in human-likeness. This number steadily increases up 26% as we add more and more attention guidance. While adding attention-guidance never degrades the sequence classification performance, we need HAMs for at least half of the training data to see a significant improvement in accuracy.

6.3.5.4 HUG-LSTM performance with a Limited Budget

Based on our experimental evaluation, the HUG model shows great potential as it can maintain the accuracy of the primary task no matter how much guidance is



(a) Accuracy



(b) Human-likeness

Figure 6.4: Varying amount of data used for attention regularization. We use 100% of the reviews for classification and a varying percentage of HAMs (x axis) for attention guidance. For the baseline experiment, the amount of data is fixed (no guidance).

Table 6.5: We assign a fixed budget for data collection and we split this budget for collecting primary task labels (cost=x) and word-level attention labels (cost=y). Collecting an attention label costs half of collecting a primary task label and the budget function is $x + y/2 = 1$.

	Training Reviews(%) (x)	Accompanying HAM(%) (y)	Metrics		
			Accuracy	Human-likeness	Combined Performance
Baseline	100	0	0.834 ± 0.006	0.497 ± 0.004	0.665
Budget function: x+y/2=1	90	20	0.835 ± 0.009	0.543 ± 0.004	0.687
	80	40	0.842 ± 0.002	0.685 ± 0.022	0.761
	70	60	0.845 ± 0.013	0.731 ± 0.006	0.787
	60	80	0.845 ± 0.012	0.742 ± 0.005	0.793

used while being significantly more interpretable compared to traditional attention. Further, if enough data is used, it improves both classification and human guidance performances. However, these win-win results require a HAM data collection, which can be costly. Next, we evaluate the relative benefit of spending more time on collecting primary task labels versus word-level attention labels.

To this end, we run a set of experiments where we assign a fixed budget for collecting labeled data. The first experiment assigns the same cost for collecting the primary task label (e.g., the sentiment of the review) and attention annotations for a single review. We fix the sum of the cost to 1 and spend varying amounts of the budget for collecting sentiment labels (x) and collecting attention labels (y). Hence, the budget function is $x + y = 1$. The baseline model uses 100% of its budget on collecting document-level sentiment labels as it does not use any attention guidance. Table 6.4 presents the results of these experiments. Results show that spending some of the budget on collecting HAMs instead of collecting more sentiment labels to make the training data larger result in: 1) As accurate models as the baseline even though less data is used for the sequence classification task, 2) Better human-likeness, and 3) Better overall performance. We believe that this is a highly preferable case in many domains, such as healthcare and autonomous vehicles, where interpretability is vital.

We argue that, in practice, collecting word-level annotations is even less costly than the budget function we experiment with. Because deciding on word-level importance is a sub-task of determining a document-level sentiment label. That is, the user has to complete the inner task first to complete the outer task.

Thus, we conduct a second experiment that specifies another budget function

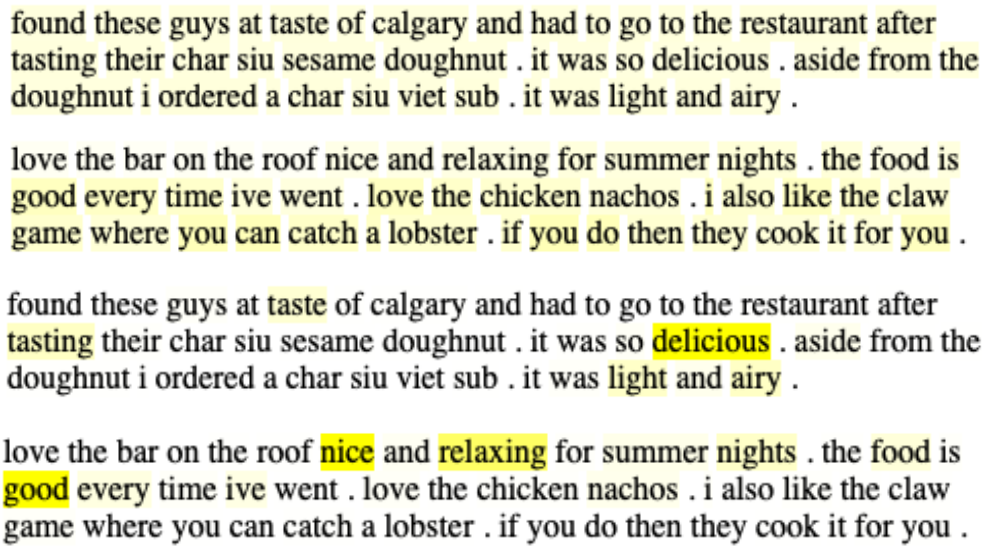


Figure 6.5: Two test examples from the YELP-HAT dataset. MAMs generated by unguided attention (top) vs. HUG-LSTM (bottom)

$x + y/2 = 1$, where collecting an attention label (y) costs half of collecting a sentiment label (x). The baseline model uses 100% of its budget on collecting document-level sentiment labels as it does not use any attention guidance. Table 6.5 presents the results of this experiment. With this budget function, attention-guided models win over baseline in all three metrics.

6.3.6 Case Study: How do attention weights change with human guidance

We conduct a case study where we qualitatively evaluate how the attention weights change with human guidance. Figure 6.5 showcase two different reviews from the test set of the YELP-HAT dataset. Machine attention maps are generated by unguided attention in the top two reviews in Figure 6.5, and by HUG-LSTM for the bottom two reviews. These MAMs depict that, unguided attention is far from being "explanation" about why the model classifies these reviews as positive or negative. On the other hand, MAMs generated by HUG model provide clues about the reasoning of the classification decision from the first glance. For example, the first review is classified as positive because the food is great at this restaurant.

The second review is classified as positive since the atmosphere there is nice and relaxing.

Improved interpretability can be even more important in a medical setting. For example, physicians would be able to tell why a particular patient is being classified as a heart disease patient by looking at the guided attention weights.

6.3.7 Summary

In this work, we propose a novel explainable deep learning framework. This framework contains a human-guided attention mechanism, HUG, and a complementary learning scheme that allows guiding attention weights with human intuition while also optimizing for a primary classification task. Instead of learning the attention scores unsupervised as in traditional attention mechanisms, it employs a learning paradigm where the model is penalized as its attention scores differ from human attention. Thus, attention scores correspond to human-like reasonings for the classification decision. Our proposed HUG framework is general and is capable of being paired with many different deep sequential models, such as RNN or BERT. It is also scalable, in that it can either conduct attention guidance for the whole training dataset or also regularization of the attention function if only partial attention guidance data is available. Through an extensive experimental evaluation, we demonstrate that attention supervision with human attention data makes the model more accurate and more interpretable concurrently. We further show that even small amounts of human guidance data are effective for driving the attention scores closer to human attention and improving accuracy.

This task has resulted in following manuscript:

- C. Sen, T. Hartvigsen, D. Zhang, J. Thadajarassiri, X. Kong, E. Rundensteiner, “Explainable Document Classification with Human-guided Attention”, In Submission to ICDM 2020.

Chapter 7

Related Work

7.1 Classifying Documents and Document Series

7.1.1 Clinical Note Classification

Text is one of the most prevalent data types in Electronic Health Records, with some examples including nursing progress reports, discharge summaries, and results of medical procedures. Previous works utilize this unstructured text to build patient-level predictive models. A large body of work uses bag-of-words representations followed by linear classification methods such as SVM [10, 44, 76]. In [31] and [32], authors examine latent variable models, namely LDA, to decompose free-text notes into features for mortality prediction. They divide the hospital stay of a patient into time windows and then extract features from aggregated notes within each time window. LDA and topic modeling techniques are used in other studies including for intervention prediction [100] and for readmission prediction [89]. [32] uses multi-task Gaussian Processes for multivariate time series modeling, incorporating both physiological signals and clinical notes. They transform a variety of irregularly-sampled clinical data into a new latent space using the hyper-parameters of multi-task Gaussian Processes models. In [11], noun-based, term-based, and topic-based features are extracted from clinical notes for named-entities through medical dictionaries such as SNOMED [95].

More recently, [27] embraces two deep learning approaches for learning representations from clinical notes. The first approach uses GloVe [73] to learn

low-dimensional dense embeddings of clinical terms. Patient-level representations are derived by aggregating the embeddings. The second approach uses an RNN with bag-of-words representations of a sequence of notes. Clinical notes of a patient bear a nested sequential data structure: the order of words is the semantic axis and the order of documents is the time axis. State-of-the-art patient-level classification ignores information from at least one of these two axes.

Another problem setting in clinical note classification is automatic ICD10 (diagnosis) coding of discharge summaries, which are a type of semi-structured clinical note [68, 77, 94]. ICD10 codes are used for billing purposes and they are manually assigned by experts. In this problem setting, the task is to assign a set of codes to a single document. In [68], authors explore attentional convolutional networks. They aim to bring interpretibility to why their model predicted each code with the help of per-label attention. Character-aware LSTM's with attention are utilized to generate sentence representations from specific subsections of discharge summaries [94]. In [77], memory networks draw from discharge summaries as well as Wikipedia documents to predict top ICD10 codes. These works aim to optimize an error-prone code-assigning process rather than assisting physicians in clinical decision support. They focus on classifying a single document, instead of document series.

More recently, BERT model [23] has achieved significant success in many NLP tasks by pre-training a deep bidirectional representation on unlabeled text, jointly conditioned on both left and right contexts. BERT architecture takes the context and the order of words into account. Owing to this success, variants of the BERT model has been proposed for the clinical domain. In particular, ClinicalBERT [**?**], an application of the BERT model to the clinical domain, is pre-trained on clinical notes from the MIMIC dataset [**?**]. Since trasformer-based models impose a length constraint on the input text, ClinicalBERT splits clinical notes into equal-length chunks and makes a prediction for each chunk. The prediction for the patient is then an aggregation of predicted values from each chunk. This approach does not consider the creation time of each clinical note. Further, it also ignores the multi-level sequential information in the sequence of clinical notes. These more recent architectures should be explored for the problems solved in this dissertation.

7.1.2 Attention-based Networks for Classifying Note Series

Attention in the natural language processing domain was originally introduced for neural translation task [5]. Since then, Recurrent Neural Networks with attention mechanisms have become state-of-the-art for diverse tasks including document classification [114, 119] and temporal EHR mining [15, 16, 65]. In medicine, accuracy and interpretability are the two most prominent factors while designing predictive models. RETAIN [16] first proposed using attention with RNN to bring interpretability to complex neural networks without sacrificing any prediction power for EHR modeling. GRAM [15] is a graph-based attention model for clinical representation learning which uses medical ontologies to learn representations and an RNN to model patient visits. Dipole [65] experiments with three types of attention mechanisms: (i) location-based, (ii) general, and (iii) concatenation-based, to compute the attention weights. In all these works, attention weights are computed based only on the EHR data, that is, the attention mechanism is not *time-informed*.

Hierarchical attention networks (HAN), another common architecture for text data, were first introduced in [114], motivated by the structure of a single document. In this work, authors showed that first focusing on the word and sentence levels individually while making a document-level classification leads to significant performance gains. Since then, HAN-based models have been applied to many problems, from question answering [17] to recommendation systems [115], and a number of variations have been proposed [30, 71, 93, 111].

HAN models attracted attention in the medical informatics domain as well. In [30], authors use hierarchical recurrent neural networks to detect cancer status given a pathology report. They utilize word and line level attention where documents are composed of lines. In [93], using visit-level medical codes (ICD, CPT) on longitudinal data, attention weights are learned for medical codes and patient visits. In other words, they use code-level and visit-level as their hierarchy. However, this model ignores the temporality across different visits of a patient.

Clinical note sequences of a patient present a similar hierarchical structure to a single document but have some significant differences. A single document consists of

words and sentences, which are written in order. Hence they are doubly-sequential. On the other hand, clinical note sequences are irregularly spaced timed series of documents, which adds a temporal dimension to the hierarchy. In addition, they have external attributes at each level of the hierarchy that need to be incorporated into the model.

7.1.3 Time-aware Deep Learning Models

Sequential deep learning algorithms assume regular sampling of their input data. However, irregular sampling is commonly observed across many domains. Exploiting the occurrence time of observations is approached as a solution to this problem and a number of models have been proposed.

In [16], authors use RNN with attention applied at the variable and visit level to predict heart failure. In this work, they propose using the time interval as an additional input feature. Some work modifies the RNN cell to account for time. For example, [118] uses a time decay term in the update gate in GRU to find a tradeoff between the previous hidden state and the candidate hidden state. [74] extends the forget gate of the standard LSTM unit to a logarithmic or cubic decay function of time intervals between two time stamps. [12] applies a time decay function to the previous hidden state in Gated Recurrent Unit (GRU) before calculating the new hidden state. [9] first decomposes memory cell in LSTM into long-term memory and short-term memory, then applies time decay to discount the short term memory, and finally calculates the new memory by combining the long-term memory and a discounted short-term memory. Main goal in these papers is to handle missing values in time series data, hence, they attempt to discount the effect of an observation if more time passed, which is not always true in our problem. In addition, modifying RNN units limits the interpretability of the resulting models, since RNN units are usually treated as black boxes.

Recently, simple time-attention mechanisms have been proposed within the spoken language understanding domain [14, 96, 97]. In these works, either a hand-picked fixed function of time [14] or a parameterized time-decay function [96] serve as the attention weights. [6] uses a disease progression function to control how much information flows into RNN at each timestemp. The input to these time functions corresponds to a scalar representation of time, namely the time difference

between instances. It is overlooked that other representations of time have the potential to be even more informative for certain tasks and/or domains – which would not be known prior to learning a model.

7.2 Interpretability of Attention-based Models

7.2.1 Interpretable Deep Learning Models

There are many efforts to design interpretable deep learning models. One way of achieving interpretability in deep learning models is to introduce interpretability constraints. Interpretability can directly be incorporated into the model structures to make models self-explanatory. The most common way of achieving this is to impose interpretability constraints on the model while being trained from data as usual. An example of this is interpretable convolutional neural networks (CNN) [116]. This model incorporates a regularization loss to higher convolutional layers of CNN to learn disentangled representations. As a result, filters can detect semantically-meaningful natural objects. One disadvantage of this design paradigm that it may create a trade-off between prediction accuracy and interpretability.

Rationale-based models [7, 57] are another type of interpretable model family, and they are used in the NLP domain. These models are trained to generate rationales and predictions from input text simultaneously. Rationales are defined as "human-like reasonings" for the model's prediction decision. Rationales are directly extracted from the input itself. They can be considered as a type of hard attention. This model consists of two main parts that are jointly learned: a generator and an encoder. The generator specifies a distribution over the input text to select candidate rationales. The encoder is used to make predictions based on the rationales. The two components are integrated and regularized in the cost function with two hyper-parameters, selection lambda, and continuity lambda, for optimizing the representative selections. The selection lambda penalizes the number of words selected, while the continuity lambda encourages the continuity via minimizing the distances of the words chosen. Thus, a higher selection lambda tends to select fewer words as rationales, while a higher continuity lambda tends to promote phrases as meaningful rationales.

7.2.2 Explainability of Attention

Attention-based models have become the architectures of choice for a vast number of NLP tasks including, but not limited to, language modeling [20], machine translation [5], document classification [114], and question answering [54, 98]. A large body of work has been using attention mechanisms to attempt to bring "interpretability" to model predictions [16, 93, 114]. While attention mechanisms have been said to add interpretability since their introduction [5], the investigation of whether this claim is correct has only just recently become a topic of high-interest [68, 92, 104]. If attention mechanisms indeed offer a more in-depth understanding of a model's inner-workings, application areas from model debugging to architecture selection would benefit significantly from profound insights into the internals of attention-based neural models.

Recently, [41], [108], and [92] proposed three distinct approaches for evaluating the explainability of attention. [41] base their work on the premise that explainable attention scores should be unique for a given prediction as well as consistent with other feature-importance measures. This leads them to conclude that attention is not explanation. Based on similar experiments on alternative attention scores, [92] conclude that attention does not necessarily correspond to the importance of inputs. In contrast, [108] find that attention learns a meaningful relationship between input tokens and model predictions which cannot be easily 'hacked' adversarially.

Whether attention equals interpretability or not depends on the definition of interpretability. Let us first assume that we define interpretability as *transparency* (as in [58]) as overall human-understanding of a model, i.e., why a model makes its decisions. Under this definition, attention scores can be seen as a vehicle of partial transparency. That is, they provide a look into the inner workings of a model, in that they produce an easily-understandable weighting of hidden states. On the other hand, whether adversarial attention scores exist that result in the same predictions as the original attention scores helps us to understand if attention is *faithful*. With their empirical analyses, [92] and [41] show that attention is not faithful. Many definitions of interpretability include *human-like explanations* for model behaviors. Evaluating the interpretability of attention from this perspective requires the collection of human attention data and an evaluation strategy for comparing machine attention to that of humans.

[21] conducted the first quantitative assessment of computational attention mechanisms for the visual question answering (VQA) task. They collect a human attention dataset, then measure the similarity of human and machine attention within the context of VQA. This VQA-HAT dataset now provides a fertile research vehicle for researchers in computer vision for studying the supervision of the attention mechanism [60]. The development of a similar dataset and an in-depth quantitative evaluation for text to advance NLP research is sorely lacking. In a concurrent and independent work, [24] collects the ERASER dataset for human annotations of rationales. While ERASER includes multiple datasets for a number of NLP tasks with relatively small amounts of data for each, we focus on text classification and collect a large amount of data on a different corpus.

7.2.3 Improving Interpretability: Supervised Attention Models

Unsupervised attention models tend to generate attention maps that do not reflect the human's intuition. This risks degrading the model's interpretability and may result in incorrect predictions [79]. Recently, researchers have searched for ways of supervising the attention mechanism to learn more accurate attention functions. Most of these works base their supervision mechanisms on manually-defined, fixed word lists of relevant words that are deemed important based on domain knowledge [63, 70, 117].

Attention supervision is also used for machine translation [52, 61]. In these works, guidance from conventional alignment models is used to supervise the attention mechanism. Some works generate weak supervision from external datasets where there is a large amount of sometimes imprecisely-labeled data [8, 13]. For example, [13] aims to generate *weak supervision* from secondary data sources. They focus on sports video analysis, where videos contain scenes of multiple people. Weak supervision gathered from sports websites is then used in the form of an action taking place in a video clip, without the specification of the person performing the action.

While these works show supervised attention can improve accuracy, guiding the attention with human data collected specifically for the primary task has not been

proposed thus far. This may, in part, be due to the lack of such human-attention map datasets in the NLP area, a challenge that is rapidly dwindling this past year [24, 91] - opening the opportunity to explore this open research question.

Chapter 8

Conclusion

In this dissertation, I study four problems.

I first propose the Attributed Hierarchical Attention model in Section 3. Clinical notes present a nested sequential structure, namely, for each patient, there is a series of free-form text documents (notes) over time and each document itself consists of a sequence of words. These notes are accompanied by external attributes at each level of granularity. State-of-the-art predictive modeling of clinical notes neglects information from at least one of these sequential axes. In this task, I propose an Attributed Hierarchical Attention network with multiple attention mechanisms conditioned on external attributes at different layers for predictive modeling of the sequence of clinical notes. I evaluate our method on three distinct clinical prediction tasks, namely, Clostridium Difficile Infection prediction, MRSA infection prediction, and in-hospital mortality prediction. Patient cohorts are extracted from the publicly-available Electronic Health Records data from Beth Israel Medical Center (MIMIC-III database). I extensively evaluate our method's prediction performance on these three tasks. I conclude that considering word-order, note-order, and the combination of the two outperform current state-of-the-art methods for clinical note classification. Moreover, external attributes are also shown to be beneficial by utilizing inferred patient profiles for achieving improved prediction. By including attention mechanisms, our method has been shown to recommend either whole-notes or specific sentences for clinicians to spend their valuable time reading.

I describe the Time-Enhanced Dual Attention model in Section 4. In this chapter,

I design a novel attention mechanism, TEND: Time Enhanced Dual Attention, and an end-to-end deep network architecture utilizing this attention mechanism, TEND-LSTM. Our TEND-LSTM model is effective for sequential document classification tasks where the input documents have associated timestamps. The TEND deep network is comprised of two attention layers, the first layer is to learn content based attention for the document sequence and the second layer is to learn a task-specific combination of content and time.I evaluate the performance of the TEND-LSTM model using six real-world clinical note datasets. Empirical results show that TEND-LSTM outperforms strong baselines and state-of-the-art methods.

After designing these attention-based classification algorithms, I turn my focus into model interpretability. Section 5 describes our data collection study for human attention maps, and our analysis for comparing human attention to machine-generated attention. To gain a deeper understanding of the relationships between human and attention-based neural network models, I conduct a large crowd-sourcing study to collect human attention maps for text classification. This human attention dataset represents a valuable community resource that I then leverage for quantifying similarities between human and attention-based neural network models using novel attention-map similarity metrics. Our research not only results in insights into significant similarities between bidirectional RNNs and human attention, but also opens the avenue for promising future research directions.

In Section 6, I propose a novel explainable deep learning framework. This framework contains a human-guided attention mechanism, HUG, and a complementary learning scheme that allows guiding attention weights with human intuition while also optimizing for a primary classification task. Instead of learning the attention scores unsupervised as in traditional attention mechanisms, it employs a learning paradigm where the model is penalized as its attention scores differ from human attention. Thus, attention scores correspond to human-like reasonings for the classification decision. Our proposed HUG framework is general and is capable of being paired with many different deep sequential models, such as RNN or BERT. It is also scalable, in that it can either conduct attention guidance for the whole training dataset or also regularization of the attention function if only partial attention guidance data is available. Through an extensive experimental evaluation,I demonstrate that attention supervision with human attention data

makes the model more accurate and more interpretable concurrently. I further show that even small amounts of human guidance data are effective for driving the attention scores closer to human attention and improving accuracy.

Next, I discuss many interesting avenues for further discovery in this area.

Chapter 9

Future Directions

9.1 Classifying Hierarchical Text Data with Time Element

In this dissertation, we mainly explore recurrent models with external attention mechanisms for classifying document series. However, transformer-based architectures [105] have recently gained an edge over recurrent architectures for many NLP tasks [23, 56, 80, 105, 113].

One promising direction is to use transformer-based architectures and BERT in a hierarchical setting similar to HAC-RNN to model clinical notes. These algorithms have been used for text and document classification. However, using them for classifying document series and timed document series requires custom-tailored design owing to the same challenges we explained in Chapters 3 and 4.

Below, I provide a brief description of the potential model design. Since BERT-based models enforce a length constraint on the input text, this model should split notes into equal length subsequences ("chunks"). The model design should take the interrelations among chunks and notes into account, and it should leverage both the time and multi-level sequential information inherent in clinical notes.

A bottom-up architecture would have four basic layers corresponding to the main tasks of the hierarchy. The bottommost layer encodes the text of each chunk into a content embedding utilizing a transformer-encoder layer. The next layer merges each content embedding and sequential information of both the note

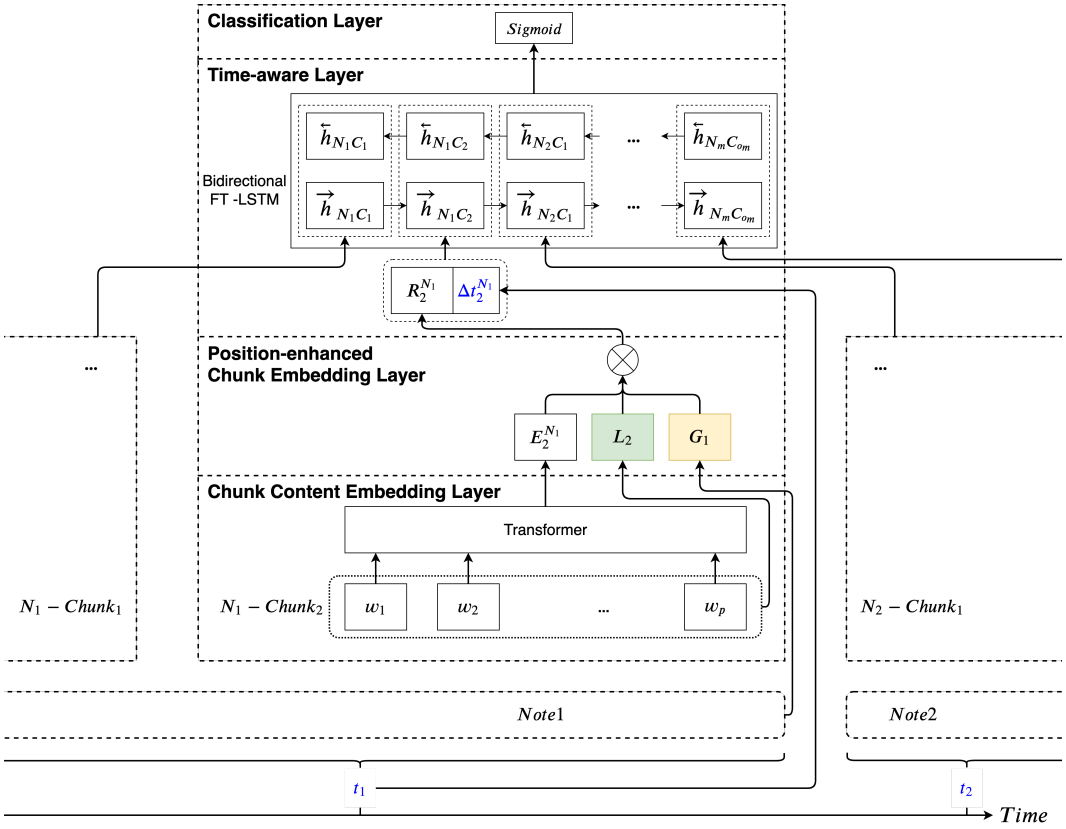


Figure 9.1: A BERT-based hierarchical model similar to HAC-RNN proposed in Chapter 3, also incorporating a time-aware layer similar to TEND-LSTM proposed in Chapter 4.

and its contained chunk into a single representation. This layer is followed by a time-aware layer to incorporate temporal information for generating heuristic patient representations. Finally, the topmost is a classification layer generating a patient-level prediction using this learned patient representation. This architectures is presented in Figure 9.1.

9.2 Towards More Interpretable Deep Learning

9.2.1 Learning from Multiple Annotators

In our YELPHAT dataset, we collect human attention maps from multiple annotators for each review. Further, many different annotators participated in the

data collection study. Some of these annotators may provide higher quality data, whereas some are less careful and less reliable. One promising future direction can be focusing on learning confidence weights for each annotator [86, 112].

This can be achieved by assigning weights to each annotator proportional to their accuracies in labeling the ground truth. We can then estimate the ground truth human attention map based on the provided data and these annotator-level confidence scores. A potential challenge may be the dependence of the annotator accuracies on the input instances (i.e., the same annotator providing varying quality of data based on the easiness of the particular data input).

9.2.2 Human Attention for Transformers

In Chapter 6, we explore BERT as one of the core sequence models in our HUG Framework. In our design, we pair BERT with an external attention layer to make it more consistent with other architectures we experiment with. However, this is a slightly artificial approach, as the BERT model already incorporates many intrinsic attention layers. Human supervision can be explored for these intrinsic attention layers.

In [19], authors explore the distribution of attention weights depending on a given input for BERT's scaled dot product self-attention. They analyze if certain attention layers are responsible for specific language understanding tasks. Based on the conclusions they reach, human supervision can be utilized to guide appropriate attention layers.

With standard attention mechanisms, the attention scores are used to compute a weighted sum of token representations.

$$c = \sum \alpha_i * h_i \tag{9.1}$$

However, as attention scores are multiplied by the learned token representations, tokens with high attention weights do not necessarily contribute much to the output [51]. An alternative approach can be learning the attention weights directly on the input text or training a model to identify spans of the input text which support its prediction [53, 69, 82]. One can build a model that can identify relevant words and let these words make the highest contribution to solving the prediction task accurately.

9.2.3 Weak Supervision from Human Attention Maps

In this dissertation, we explore ways to fully supervising the attention mechanism. However, another approach can be weak supervision. Weak supervision could be preferred if full supervision causes a deterioration in models predictive performance or if the human attention map dataset is not large enough for full supervision.

Existing literature has explored learning to generate "human-like attention" to utilize this synthetic data for attention supervision for visual question answering task [79]. Similar strategies can be used for generating human attention maps using our YELP-HAT dataset. This enhanced dataset (i.e., a combination of the real data collected from human annotators and the synthetic data) can be used for attention supervision.

One potential limitation of the YELP-HAT dataset is that, as it is collected via crowd-sourcing, not every collected annotation is of equal quality. Some data instances, thus, may create noise in the label set. Machine learning methods designed for dealing with learning from noisy labels could be another promising direction for better attention supervision.

Finally, instead of *Human-gudied attention*, one can explore *Human-inspired attention*. By analyzing collected human attention maps from different perspectives, we design constraints on the attention scores to make machine-learned attention more human-like in indirect ways. For example, if human annotators tend to pick consecutive words, a continuity parameter for high attention words can be incorporated. One observation we made during our analysis in Chapter 5 was that human annotators select adjectives more frequently when deciding a sentiment. Similarly, machine attention can be forced to assign higher attention scores on adjectives.

Bibliography

[1] National nlp clinical challenges (n2c2).

[2] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016.

[3] T. Alkhouli, G. Bretschner, J.-T. Peter, M. Hethnawi, A. Guta, and H. Ney. Alignment-based neural machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 1, Research Papers*, pages 54–65, 2016.

[4] M. Auli, M. Galley, C. Quirk, and G. Zweig. Joint language and translation modeling with recurrent neural networks. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1044–1054, 2013.

[5] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *ICLR*, 2015.

[6] T. Bai, S. Zhang, B. L. Egleston, and S. Vucetic. Interpretable representation learning for healthcare via capturing disease progression through time. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 43–51. ACM, 2018.

[7] Y. Bao, S. Chang, M. Yu, and R. Barzilay. Deriving machine attention from human rationales. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1903–1913, 2018.

[8] M. Barrett, J. Bingel, N. Hollenstein, M. Rei, and A. Søgaard. Sequence classification with human attention. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 302–312, 2018.

[9] I. M. Baytas, C. Xiao, X. Zhang, F. Wang, A. K. Jain, and J. Zhou. Patient subtyping via time-aware lstm networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 65–74. ACM, 2017.

[10] R. J. Byrd, S. R. Steinhubl, J. Sun, S. Ebadollahi, and W. F. Stewart. Automatic identification of heart failure diagnostic criteria, using text analysis of clinical notes from electronic health records. *International Journal of Medical Informatics*, 83(12):983–992, 2014.

[11] K. L. Caballero Barajas and R. Akella. Dynamically modeling patient's health state from electronic medical records: A time series approach. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 69–78. ACM, 2015.

[12] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu. Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, 8(1):6085, 2018.

[13] L. Chen, M. Zhai, and G. Mori. Attending to distinctive moments: Weakly-supervised attention models for action localization in video. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 328–336, 2017.

[14] P.-C. Chen, T.-C. Chi, S.-Y. Su, and Y.-N. Chen. Dynamic time-aware attention to speaker roles and contexts for spoken language understanding. In *IEEE Automatic Speech Recognition and Understanding Workshop*, pages 554–560, 2017.

[15] E. Choi, M. T. Bahadori, L. Song, W. F. Stewart, and J. Sun. Gram: graph-based attention model for healthcare representation learning. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 787–795. ACM, 2017.

[16] E. Choi, M. T. Bahadori, J. Sun, J. Kulas, A. Schuetz, and W. Stewart. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. In *Advances in Neural Information Processing Systems*, pages 3504–3512, 2016.

[17] E. Choi, D. Hewlett, A. Lacoste, I. Polosukhin, J. Uszkoreit, and J. Berant. Hierarchical question answering for long documents. *arXiv preprint arXiv:1611.01839*, 2016.

[18] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[19] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning. What does bert look at? an analysis of bert's attention. *arXiv preprint arXiv:1906.04341*, 2019.

[20] M. Daniluk, T. Rocktäschel, J. Welbl, and S. Riedel. Frustratingly short attention spans in neural language modeling. *ICLR*, 2017.

[21] A. Das, H. Agrawal, L. Zitnick, D. Parikh, and D. Batra. Human attention in visual question answering: Do humans and deep networks look at the same regions? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 932–937, 2016.

[22] T. Desautels, J. Calvert, J. Hoffman, M. Jay, Y. Kerem, L. Shieh, D. Shimabukuro, U. Chettipally, M. D. Feldman, C. Barton, et al. Prediction of sepsis in the intensive care unit with minimal electronic health record data: a machine learning approach. *JMIR Medical Informatics*, 4(3), 2016.

[23] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[24] J. DeYoung, S. Jain, N. F. Rajani, E. Lehman, C. Xiong, R. Socher, and B. C. Wallace. Eraser: A benchmark to evaluate rationalized nlp models. *arXiv preprint arXiv:1911.03429*, 2019.

[25] F. Doshi-Velez and B. Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

[26] M. Du, N. Liu, and X. Hu. Techniques for interpretable machine learning. *Communications of the ACM*, 63(1):68–77, 2019.

[27] S. Dubois, D. C. Kale, N. Shah, and K. Jung. Learning effective representations from clinical notes. *arXiv preprint arXiv:1705.07025*, 2017.

[28] S. El Hihi and Y. Bengio. Hierarchical recurrent neural networks for long-term dependencies. In *Advances in neural information processing systems*, pages 493–499, 1996.

[29] J. L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990.

[30] S. Gao, M. T. Young, J. X. Qiu, H.-J. Yoon, J. B. Christian, P. A. Fearn, G. D. Tourassi, and A. Ramanthan. Hierarchical attention networks for information extraction from cancer pathology reports. *Journal of the American Medical Informatics Association*, 25(3):321–330, 2017.

[31] M. Ghassemi, T. Naumann, F. Doshi-Velez, N. Brimmer, R. Joshi, A. Rumshisky, and P. Szolovits. Unfolding physiological state: Mortality modelling in intensive care units. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 75–84. ACM, 2014.

[32] M. Ghassemi, M. A. Pimentel, T. Naumann, T. Brennan, D. A. Clifton, P. Szolovits, and M. Feng. A multivariate timeseries modeling approach to severity of illness assessment and forecasting in icu with sparse, heterogeneous clinical data. In *AAAI Conference on Artificial Intelligence*, pages 446–453, 2015.

[33] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

[34] A. Graves, A.-r. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013.

[35] A. Graves and J. Schmidhuber. Offline handwriting recognition with multi-dimensional recurrent neural networks. In *Advances in neural information processing systems*, pages 545–552, 2009.

[36] A. Graves, G. Wayne, and I. Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

[37] T. Hartvigsen, C. Sen, S. Brownell, E. Teeple, X. Kong, and E. A. Rundensteiner. Early prediction of mrsa infections using electronic health records. In *ICHI*, pages 156–167, 2018.

[38] C. D. V. Hoang, T. Cohn, and G. Haffari. Incorporating side information into recurrent neural network language models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1250–1255, 2016.

[39] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[40] M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM, 2004.

[41] S. Jain and B. C. Wallace. Attention is not explanation. 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)*, pages 3543–3556, 2019.

[42] P. B. Jensen, L. J. Jensen, and S. Brunak. Mining electronic health records: towards better research applications and clinical care. *Nature Reviews Genetics*, 13(6):395, 2012.

[43] Y. Jo, N. Loghmanpour, and C. P. Rosé. Time series analysis of nursing notes for mortality prediction via a state transition topic model. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1171–1180. ACM, 2015.

[44] E. Joffe, E. J. Pettigrew, J. R. Herskovic, C. F. Bearden, and E. V. Bernstam. Expert guided natural language processing using one-class classification. *Journal of the American Medical Informatics Association*, 22(5):962–966, 2015.

[45] A. E. Johnson, T. J. Pollard, and R. G. Mark. Reproducibility in critical care: a mortality prediction case study. In *Machine Learning for Healthcare Conference*, pages 361–376, 2017.

[46] A. E. Johnson, T. J. Pollard, L. Shen, H. L. Li-wei, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, and R. G. Mark. Mimic-iii, a freely accessible critical care database. *Scientific Data*, 3:160035, 2016.

[47] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015.

[48] M. H. F. Keller. Modeling human reading with neural attention. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, volume 85, page 95, 2016.

[49] Y.-B. Kim, D. Kim, A. Kumar, and R. Sarikaya. Efficient large-scale neural domain classification with personalized attention. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2214–2224, 2018.

[50] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[51] G. Kobayashi, T. Kuribayashi, S. Yokoi, and K. Inui. Attention module is not only a weight: Analyzing transformers with vector norms. *arXiv preprint arXiv:2004.10102*, 2020.

[52] S. Kuang, J. Li, A. Branco, W. Luo, and D. Xiong. Attention focusing for neural machine translation by bridging source and target embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1767–1776, 2018.

[53] S. Kumar and P. Talukdar. Nile: Natural language inference with faithful natural language explanations. *arXiv preprint arXiv:2005.12116*, 2020.

[54] S. Kundu and H. T. Ng. A question-focused multi-factor attention network for question answering. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[55] S. M. Lakew, M. Cettolo, and M. Federico. A comparison of transformer and recurrent neural networks on multilingual neural machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 641–652, 2018.

[56] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

[57] T. Lei, R. Barzilay, and T. Jaakkola. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 107–117, 2016.

[58] Z. C. Lipton. The mythos of model interpretability. *Communications of the ACM*, 61(10):36–43, 2018.

[59] Z. C. Lipton, D. C. Kale, C. Elkan, and R. Wetzel. Learning to diagnose with lstm recurrent neural networks. In *MLHC*, page 253–270, 2016.

[60] C. Liu, J. Mao, F. Sha, and A. Yuille. Attention correctness in neural image captioning. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[61] L. Liu, M. Utiyama, A. Finch, and E. Sumita. Neural machine translation with supervised attention. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3093–3102, 2016.

[62] P. Liu, X. Qiu, and X. Huang. Recurrent neural network for text classification with multi-task learning. In *IJCAI*, pages 2873–2879. AAAI Press, 2016.

[63] S. Liu, Y. Chen, K. Liu, and J. Zhao. Exploiting argument information to improve event detection via supervised attention mechanisms. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1789–1798, 2017.

[64] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[65] F. Ma, R. Chitta, J. Zhou, Q. You, T. Sun, and J. Gao. Dipole: Diagnosis prediction in healthcare via attention-based bidirectional recurrent neural networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1903–1911. ACM, 2017.

[66] K. Marimuthu and S. L. Devi. How human analyse lexical indicators of sentiments-a cognitive analysis using reaction-time. In *Proceedings of the 2nd Workshop on Sentiment Analysis where AI meets Psychology*, pages 81–90, 2012.

[67] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.

[68] J. Mullenbach, S. Wiegreffe, J. Duke, J. Sun, and J. Eisenstein. Explainable prediction of medical codes from clinical text. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 1101–1111, 2018.

[69] S. Narang, C. Raffel, K. Lee, A. Roberts, N. Fiedel, and K. Malkan. Wt5?! training text-to-text models to explain their predictions. *arXiv preprint arXiv:2004.14546*, 2020.

[70] M. Nguyen and T. Nguyen. Who is killed by police: Introducing supervised attention for hierarchical lstms. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2277–2287, 2018.

[71] N. Pappas and A. Popescu-Belis. Multilingual hierarchical attention networks for document classification. *arXiv preprint arXiv:1707.00896*, 2017.

[72] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318, 2013.

[73] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[74] T. Pham, T. Tran, D. Phung, and S. Venkatesh. Deepcare: A deep dynamic memory model for predictive medicine. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 30–41. Springer, 2016.

[75] R. Pirracchio, M. L. Petersen, M. Carone, M. R. Rigon, S. Chevret, and M. J. van der Laan. Mortality prediction in intensive care units with the super icu learner algorithm (sicula): a population-based study. *The Lancet Respiratory Medicine*, 3(1):42–52, 2015.

[76] C. Poulin, B. Shiner, P. Thompson, L. Vepstas, Y. Young-Xu, B. Goertzel, B. Watts, L. Flashman, and T. McAllister. Predicting the risk of suicide by analyzing the text of clinical notes. *PloS One*, 9(1):e85733, 2014.

[77] A. Prakash, S. Zhao, S. A. Hasan, V. V. Datla, K. Lee, A. Qadir, J. Liu, and O. Farri. Condensed memory networks for clinical diagnostic inferencing. In *AAAI*, pages 3274–3280, 2017.

[78] S. Pyysalo, F. Ginter, H. Moen, T. Salakoski, and S. Ananiadou. Distributional semantics resources for biomedical text processing. *Proceedings of LBM*, pages 39–44, 2013.

[79] T. Qiao, J. Dong, and D. Xu. Exploring human-like attention supervision in visual question answering. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[80] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*, 2018.

[81] C. Raffel and D. P. Ellis. Feed-forward networks with attention can solve some long-term memory problems. *arXiv preprint arXiv:1512.08756*, 2015.

[82] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

[83] W. Raghupathi and V. Raghupathi. Big data analytics in healthcare: promise and potential. *Health information science and systems*, 2(1):3, 2014.

[84] A. Rajkomar, E. Oren, K. Chen, A. M. Dai, N. Hajaj, M. Hardt, P. J. Liu, X. Liu, J. Marcus, M. Sun, et al. Scalable and accurate deep learning with electronic health records. *NPJ Digital Medicine*, 1(1):18, 2018.

[85] M. O. Riedl. Human-centered artificial intelligence and machine learning. *arXiv preprint arXiv:1901.11184*, 2019.

[86] F. Rodrigues and F. C. Pereira. Deep learning from crowds. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[87] C. Rudin. Please stop explaining black box models for high stakes decisions. *arXiv preprint arXiv:1811.10154*, 2018.

[88] C. Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

[89] A. Rumshisky, M. Ghassemi, T. Naumann, P. Szolovits, V. Castro, T. McCoy, and R. Perlis. Predicting early psychiatric readmission with natural language processing of narrative discharge summaries. *Translational Psychiatry*, 6(10):e921, 2016.

[90] C. Sen, T. Hartvigsen, E. Rundensteiner, and K. Claypool. Crest-risk prediction for clostridium difficile infection using multimodal data mining. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 52–63. Springer, 2017.

[91] C. Sen, T. Hartvigsen, B. Yin, X. Kong, and E. Rundensteiner. Human attention maps for text classification: Do humans and neural networks focus on the same words? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[92] S. Serrano and N. A. Smith. Is attention interpretable? 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)*, 2019.

[93] Y. Sha and M. D. Wang. Interpretable predictions of clinical outcomes with an attention-based recurrent neural network. In *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 233–240. ACM, 2017.

[94] H. Shi, P. Xie, Z. Hu, M. Zhang, and E. P. Xing. Towards automated icd coding using deep learning. *arXiv preprint arXiv:1711.04075*, 2017.

[95] K. A. Spackman, K. E. Campbell, and R. A. Côté. Snomed rt: a reference terminology for health care. In *AMIA Annual Fall Symposium*, page 640. American Medical Informatics Association, 1997.

[96] S.-Y. Su, P.-C. Yuan, and Y.-N. Chen. How time matters: Learning time-decay attention for contextual spoken language understanding in dialogues. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 2133–2142, 2018.

[97] S.-Y. Su, P.-C. Yuan, and Y.-N. Chen. Learning context-sensitive time-decay attention for role-based dialogue modeling. *arXiv preprint arXiv:1809.01557*, 2018.

[98] S. Sukhbaatar, J. Weston, R. Fergus, et al. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448, 2015.

[99] Y. Sun, S. Wang, Y. Li, S. Feng, H. Tian, H. Wu, and H. Wang. Ernie 2.0: A continual pre-training framework for language understanding. *arXiv preprint arXiv:1907.12412*, 2019.

[100] H. Suresh, N. Hunt, A. Johnson, L. A. Celi, P. Szolovits, and M. Ghassemi. Clinical intervention prediction and understanding with deep neural networks. In *Machine Learning for Healthcare Conference*, pages 322–337, 2017.

[101] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[102] D. Tang, B. Qin, and T. Liu. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1422–1432, 2015.

[103] Y. Tay, M. C. Phan, L. A. Tuan, and S. C. Hui. Learning to rank question answer pairs with holographic dual lstm architecture. In *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*, pages 695–704, 2017.

[104] J. Thorne, A. Vlachos, C. Christodoulopoulos, and A. Mittal. Generating token-level explanations for natural language inference. 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)*, pages 963–969, 2019.

[105] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[106] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164, 2015.

[107] Y. Wang, M. Huang, X. Zhu, and L. Zhao. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 606–615, 2016.

[108] S. Wiegreffe and Y. Pinter. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, 2019.

[109] J. Wiens, E. Horvitz, and J. V. Guttag. Patient risk stratification for hospital-associated c. diff as a time-series classification task. In *Advances in Neural Information Processing Systems*, pages 467–475, 2012.

[110] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[111] C. Xing, Y. Wu, W. Wu, Y. Huang, and M. Zhou. Hierarchical recurrent attention network for response generation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[112] Y. Yan, R. Rosales, G. Fung, R. Subramanian, and J. Dy. Learning from multiple annotators with varying expertise. *Machine learning*, 95(3):291–327, 2014.

[113] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764, 2019.

[114] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, 2016.

[115] H. Ying, F. Zhuang, F. Zhang, Y. Liu, G. Xu, X. Xie, H. Xiong, and J. Wu. Sequential recommender system based on hierarchical attention networks. In *the 27th International Joint Conference on Artificial Intelligence*, 2018.

[116] Q. Zhang, Y. Nian Wu, and S.-C. Zhu. Interpretable convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8827–8836, 2018.

[117] Y. Zhao, X. Jin, Y. Wang, and X. Cheng. Document embedding enhanced event detection with hierarchical and supervised attention. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 414–419, 2018.

[118] K. Zheng, W. Wang, J. Gao, K. Y. Ngiam, B. C. Ooi, and W. L. J. Yip. Capturing feature-level irregularity in disease progression modeling. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1579–1588. ACM, 2017.

[119] X. Zhou, X. Wan, and J. Xiao. Attention-based lstm network for cross-lingual sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 247–256, 2016.