AMAS: Attention Model for Attributed Sequence Classification

Zhongfang Zhuang* Xiangnan Kong* Elke Rundensteiner*

Abstract

Classification over sequential data is important for a wide range of applications from information retrieval, anomaly detection to genomic analysis. Neural network approaches, in particular recurrent neural networks, have been widely used in such tasks due to their strong capability of feature learning. However, recent innovations in sequence classification learn from not only the sequences but also the associated attributes, called *attributed sequences*. While recent work shows the attributed sequences to be useful in real-world applications, neural attention models have not yet been explored for attributed sequence classification. This paper is the first to study the problem of attributed sequence classification with the neural attention mechanism. This is challenging that now we need to assess the importance of each item in each sequence considering both the sequence itself and the associated metadata. We propose a framework, called `AMAS`, to classify attributed sequences using the information from the sequences, metadata, and the computed attention. Empirical results on real-world datasets demonstrate that the proposed `AMAS` framework significantly improves the performance of classification over the state-of-the-art methods on attributed sequences.

1 Introduction

Classification over sequential data is important for a wide range of applications over a variety of research areas, including document classification in information retrieval and gene classification in genomic analysis. The conventional approaches to sequence classification focus on only the sequential data, whereas the metadata associated with the sequences is often discarded. However, metadata is has been shown to be useful for a variety of topics [30, 26]. In the literature, the data type of a sequence and its associated metadata as a unity is called *attributed sequence*. Attributed sequences are heterogeneously structured and thus not naturally represented as fixed-size vectors. For example, genes can be represented as attributed sequences, where each gene consists of a DNA *sequence* and a *set* of *attributes* (*e.g.,* PPI, gene ontology) indicating the properties of the gene. Network traffic can also be modeled as attributed sequences. Namely, each network transmission session consists of a sequence of packages being sent (or received) by a router and a set of attributes characterizing the network traffic, (*e.g.,* date, time, size sender priority *etc*).

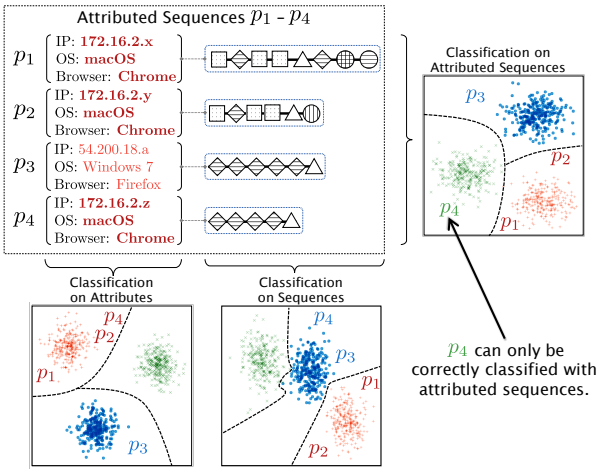


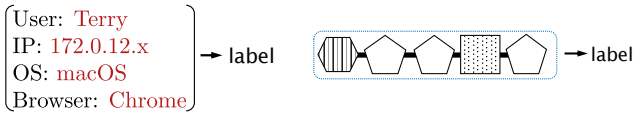
Figure 1: Using **both** attributes and sequences is essential for predicting the label of p_4.

Designing a classifier to correctly predict the class label of an attributed sequence is beneficial for applications such as automatic network traffic profiling for cyber security. However, it has been shown that it is challenging to classify sequences by themselves [31, 24, 19, 25]. With dependencies between attributes and sequences, the attributed sequence classification must tackle several important design challenges.

In this paper, we study the problem of attributed sequence classification. This problem is different from prior sequence classification work [24, 19] as we now need to extract feature vectors from not only the *sequences* but also incorporating the metadata as the *attributes*, along with the *dependencies* between attributes and sequences. We summarize the challenges below.

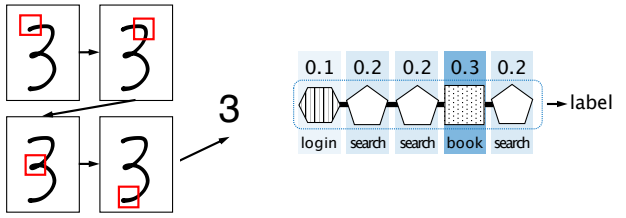
- Attribute-sequence dependencies. Contrary to the simplifying assumption that the attributes and sequences are independent, there are various dependencies between them. Dependencies between at-

*{zzhuang, xkong, rundenst}@wpi.edu, Computer Science Department, Worcester Polytechnic Institute



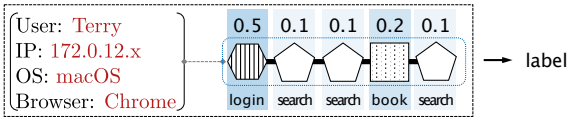
(a) Classification on data attributes [1].

(b) Sequence classification [31].



(c) Image classification with attention [20].

(d) Sequence classification with attention [28].



(e) Sequence classification with attribute-guided attention. (this paper)

Figure 2: A comparison of related classification problems.

tributes and sequences raise problems when learning to classify attributed sequence data. For example, in network traffic data, the device type of the router(*i.e.,* an attributes) may affect the pattern of sending/receiving TCP/UDP packets (*i.e.,* sequences). Since conventional sequence classification approaches focus on a single data type only, these *dependencies* would thus not be captured.

- Attention from both attributes and sequences. Recent sequence learning research [34] has used neural attention models to improve the performance of sequence learning, such as document classification [34]. However, the attention mechanism focuses on learning the weight of certain time steps or sub-sequences in each sequential instance, without regards to its associated attributes. With information from the attributes, the weight of item or sub-sequence may be drastically different from the weight calculated by the attention mechanism using only sequences, which would consequently have different classification results.

To address the above challenges, we propose an end-to-end solution for attributed sequence classification with an attention model, called `AMAS`. The `AMAS` model includes three main components: a `Attribute Net` to encode the information from attributes, a `Sequence Net` to encode the information from sequences, and an `Attention Block` learning the attention from not only the sequences but also the hybrid of information from both attributes and sequences. Our paper offers the following contributions:

- We formulate the problem of attributed sequence classification.
- We design a deep learning framework, called `AMAS`, with two attention-based models to exploit the information from attributes and sequences.
- We demonstrate that the proposed models significantly improve the performance of attributed sequence classification using performance experiments and case studies.

2 Problem Formulation

2.1 Preliminaries. We introduce the key definitions in this section. The important notations are summarized in Table 1.

Definition 1. (Sequence) Given a finite set $\mathcal{I}$ composed of r categorical items, a sequence $s_i = \left(x_i^{(1)}, \cdots, x_i^{(t_i)}\right)$ is an ordered list of t_i items, where $\forall x_i^{(t)} \in \mathcal{I}$.

We use the subscript i to distinguish different sequence instances. Following common pre-processing steps for deep learning, we apply *zero-padding* on the sequences of variable-length, in which each sequence is padded to the maximum length of the sequences in a dataset using 0's. The second step is to *one-hot encode* each sequence s_i [10]. We denote the one-hot encoded form of sequence s_i as a matrix $\mathbf{s}_i = \left[\mathbf{x}_i^{(1)}, \cdots, \mathbf{x}_i^{(t_i)}\right]$. Learning models are capable of disregarding the padding so that the padding has no effect on the training of models.

Definition 2. (Attributed Sequence) Given a u-dimensional attribute vector $\mathbf{v}_i$ is composed of u attributes, an attributed sequence p_i is a pair composed of an attribute vector $\mathbf{v}_i$ and a one-hot encoded sequence $\mathbf{s}_i$, denoted as $p_i = (\mathbf{v}_i, \mathbf{s}_i)$.

2.2 Problem Definition. We formulate the attributed sequence classification problem as the problem of finding the parameters θ of a predictor Θ that minimizes the prediction error of class labels. Intuitively, we want to maximize the possibility of correctly predicting labels when given a training set $\mathbb{P} = \{p_1, \cdots, p_k\}$ of k attributed sequences. Thus, we formulate the training process as an optimization process:

$$\arg\min_{\theta} - \sum_{i} \Pr(l_i) \log \Pr\left(\Theta\left(p_i\right)\right) \tag{2.1}$$

Table 1: Important Mathematical Notations

Notation	Description
$\mathbb{R}$	The set of real numbers.
$\mathbb{P}$	A set of attributed sequences.
r	The number of all possible items in sequences.
s_i	A sequence of categorical items.
$x_i^{(t)}$	The t-th item in sequence s_i.
$t_{\max}$	The maximum length of sequences.
$\mathbf{s}_i$	A one-hot encoded sequence in the form of a matrix $\mathbf{s}_i \in \mathbb{R}^{t_{\max} \times r}$.
$\mathbf{x}_i^{(t)}$	A one-hot encoded item at t-th time step.
$\mathbf{v}_i$	An attribute vector.
p_i	An attributed sequence. *i.e.,* , $p_i = (\mathbf{v}_i, \mathbf{s}_i)$
$\boldsymbol{p}_i$	A feature vector of attributed sequence p_i.
$\boldsymbol{\mu}_i$	Attention weights.
$\boldsymbol{\alpha}_i$	Attention vector.

Our goal is to find the parameters that minimize the categorical cross-entropy loss between the predicted labels using parameters in function Θ and the true labels for all attributed sequences in the dataset.

3 Attributed Sequence Attention Mechanism

The proposed `AMAS` model has three components, one `Attribute Net` for learning the attribute information, one `Sequence Net` to learn the sequential information, and one `Attention Block` to learn the attention from both attributes and sequences.

3.1 Network Components.

3.1.1 `Attribute Net.` We build `Attribute Net` using fully connected neural network denoted as:

$$f\left(\mathbf{A}; \mathbf{W_r}, \mathbf{b_r}\right) = \tanh(\mathbf{W}_{\mathrm{r}}\mathbf{A} + \mathbf{b}_{\mathrm{r}}) \tag{3.2}$$

where $\mathbf{W}_a$ and $\mathbf{b}_a$ are two trainable parameters in the `Attribute Net`, denoting the weight matrix and bias vector, respectively. We use the activation function tanh here in our `Attribute Net` based on our empirical studies. Other choices, such as `ReLu` or `sigmoid`, may work equally well in other real-world scenarios.

When given an attributed sequence $p_i = (\mathbf{v}_i, \mathbf{s}_i)$, `Attribute Net` takes the attributes as input and generates an attribute vector $\boldsymbol{r}_i = f(\mathbf{v}_i; \mathbf{W_r}, \mathbf{b_r})$. Different from previous work in [4, 27] using stacked fully connected neural network as autoencoder, where the training goal is to minimize the reconstruction error, our goal of `Attribute Net` is to work together with other network components to maximize the possibility of predicting the correct labels.

3.1.2 `Sequence Net.` Different from the attributes being unordered, items in our sequences have a temporal ordering. The information about the sequences is in both the item values and the ordering of items. The temporal orderings require a model that is capable of handling the dependencies between different items. There have been extensive studies on using recurrent neural networks (RNN) to handle temporal dependencies. However, RNN suffers from the problem of exploding and vanishing gradient during the training, where the gradient value becomes too large or too small and thus the network becomes untrainable. Long Short-Term Memory (LSTM) [13] is designed as one variation and expansion of the RNN to handle such issues. LSTM is capable of "remembering" values over long time intervals by introducing additional internal variables (*i.e.,* various "gates" and "cell states"). We use an LSTM to handle the dependencies. With a variable $\mathbf{X}^{(t)}$ at time t, the `Sequence Net` can be expressed as:

$$\begin{aligned}
\mathbf{i}^{(t)} &= \sigma\left(\mathbf{W}_{\mathrm{i}}\mathbf{X}^{(t)} + \mathbf{U}_{\mathrm{i}}\mathbf{h}^{(t-1)} + \mathbf{b}_{\mathrm{i}}\right) \\
\mathbf{f}^{(t)} &= \sigma\left(\mathbf{W}_{\mathrm{f}}\mathbf{X}^{(t)} + \mathbf{U}_{\mathrm{f}}\mathbf{h}^{(t-1)} + \mathbf{b}_{\mathrm{f}}\right) \\
\mathbf{o}^{(t)} &= \sigma\left(\mathbf{W}_{\mathrm{o}}\mathbf{X}^{(t)} + \mathbf{U}_{\mathrm{o}}\mathbf{h}^{(t-1)} + \mathbf{b}_{\mathrm{o}}\right) \\
\mathbf{g}^{(t)} &= \tanh\left(\mathbf{W}_{\mathrm{c}}\mathbf{X}^{(t)} + \mathbf{U}_{\mathrm{c}}\mathbf{h}^{(t-1)} + \mathbf{b}_{\mathrm{c}}\right) \\
\mathbf{c}^{(t)} &= \mathbf{f}^{(t)} \odot \mathbf{c}^{(t-1)} + \mathbf{i}^{(t)} \odot \mathbf{g}^{(t)} \\
\mathbf{h}^{(t)} &= \mathbf{o}^{(t)} \odot \tanh\left(\mathbf{c}^{(t)}\right)
\end{aligned} \tag{3.3}$$

where $\odot$ denotes the bitwise multiplication, σ is a `sigmoid` activation function, $\mathbf{i}^{(t)}$, $\mathbf{f}^{(t)}$ and $\mathbf{o}^{(t)}$ are the internal gates of the LSTM, and $\mathbf{c}^{(t)}$ and $\mathbf{h}^{(t)}$ are the cell and hidden states of the LSTM, respectively. We denote the `Sequence Net` as:

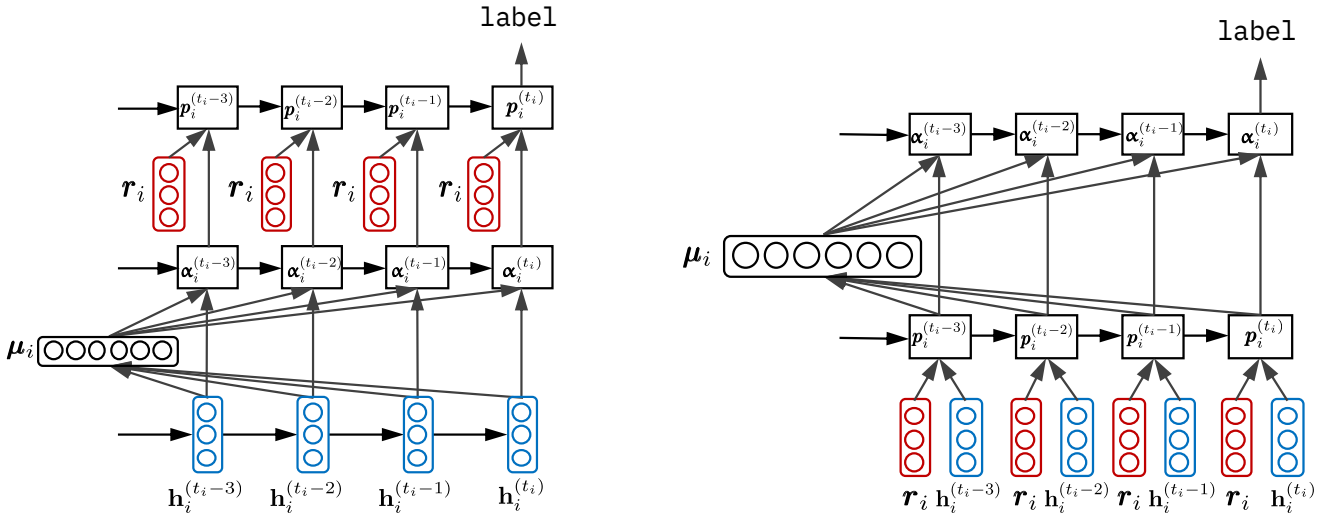
$$g(\mathbf{X}^{(t)}; \mathbf{W}_{\mathrm{s}}, \mathbf{U}_{\mathrm{s}}, \mathbf{b}_{\mathrm{s}}) = \mathbf{h}^{(t)} \tag{3.4}$$

where $\mathbf{W}_{\mathrm{s}} = [\mathbf{W}_{\mathrm{i}}, \mathbf{W}_{\mathrm{f}}, \mathbf{W}_{\mathrm{o}}, \mathbf{W}_{\mathrm{c}}]$, $\mathbf{U}_{\mathrm{s}} = [\mathbf{U}_{\mathrm{i}}, \mathbf{U}_{\mathrm{f}}, \mathbf{U}_{\mathrm{o}}, \mathbf{U}_{\mathrm{c}}]$ and $\mathbf{b}_{\mathrm{s}} = [\mathbf{b}_{\mathrm{i}}, \mathbf{b}_{\mathrm{f}}, \mathbf{b}_{\mathrm{o}}, \mathbf{b}_{\mathrm{c}}]$.

With the sequence $\mathbf{s}_i = \left[\mathbf{x}_i^{(1)}, \cdots, \mathbf{x}_i^{(t_i)}\right]$ as part of an attributed sequence p_i, the hidden states for input $\mathbf{x}_i^{(t)}$ are $g\left(\mathbf{x}_i^{(t)}; \mathbf{W}_{\mathrm{s}}, \mathbf{U}_{\mathrm{s}}, \mathbf{b}_{\mathrm{s}}\right) = \mathbf{h}_i^{(t)}$.

3.1.3 `Attention Block.` Recent work [20, 6] has identified that even LSTM-based solutions cannot fully handle the sequence learning on long sequences that the information over a long time may be lost. One popular solution to this problem is to incorporate the attention mechanism into the model. The attention mechanism effectively summarizes the data with the aim to leverage the importance of each item in the sequential input.

Attributed Sequence Attention (ASA). Different from the common sequence attention models, we now need to incorporate the attribute information into



(a) Attributed Sequence Attention (ASA). (b) Attributed Sequence Hybrid Attention (ASHA).

Figure 3: Two types of attention for attributed sequence classification in this paper, where $\boldsymbol{\mu}_i = \left[\boldsymbol{\mu}_i^{(1)}, \cdots, \boldsymbol{\mu}_i^{(t_i)}\right]$ is the attention weights.

the learning process. Here, we design the `Attention Block` as follows: First, we need to compute the attention weight $\boldsymbol{\mu}_i^{(t)}$ at t-th time as:

$$g\left(\mathbf{x}_i^{(t)}\right) = \mathbf{h}_i^{(t)}$$

$$\boldsymbol{\mu}_i^{(t)} = \frac{\exp\left(g\left(\mathbf{x}_i^{(t)}\right)\right)}{\sum_{j=1}^{t_i} g\left(\mathbf{x}_i^{(j)}\right)}$$

Then, the attention weight is multiplied with the hidden state at each time step:

$$\boldsymbol{\alpha}_i^{(t)} = \boldsymbol{\mu}_i^{(t)} \odot g\left(\mathbf{x}_i^{(t)}\right), t = 1, 2, \cdots, t_i \tag{3.5}$$

The attention weight $\mu_i^{(t)}$ at t time is randomly initialized and incrementally adjusted during the training process. The output at each time step is then augmented with the outputs from `Attribute Net` as:

$$\boldsymbol{p}_i^{(t)} = f(\mathbf{v}_i) \oplus \boldsymbol{\alpha}_i^{(t)}$$

At the last time step t_i, we denote $\boldsymbol{p}_i = \boldsymbol{p}_i^{(t_i)}$ to simply the notation.

Attributed Sequence Hybrid Attention (ASHA). Different from the previous ASA approach, the outputs of `Attribute Net` and `Sequence Net` are augmented with the `Attention Block`. The attention weight is written as:

$$d\left(\mathbf{v}_i, \mathbf{x}_i^{(t)}\right) = f(\mathbf{v}_i) \oplus g\left(\mathbf{x}_i^{(t)}\right)$$

$$\boldsymbol{\mu}_i^{(t)} = \frac{\exp\left(d\left(\mathbf{v}_i, \mathbf{x}_i^{(t)}\right)\right)}{\sum_{j=1}^{t_i} d\left(\mathbf{v}_i, \mathbf{x}_i^{(j)}\right)}$$

Then, the vectors used for classification is:

$$\boldsymbol{\alpha}_i^{(t)} = \boldsymbol{\mu}_i^{(t)} \odot d\left(\mathbf{v}_i, \mathbf{x}_i^{(t)}\right), t = 1, 2, \cdots, t_i \tag{3.6}$$

3.2 Attributed Sequence Classification.

In the solution of attributed sequence classification *without* attention, the `Attribute Net` and the `Sequence Net` are first concatenated as:

$$\boldsymbol{p}_i = d\left(\mathbf{v}_i, \mathbf{x}_i^{(t_i)}\right) = \boldsymbol{r}_i \oplus \mathbf{h}_i^{(t_i)} \tag{3.7}$$

Here, $\oplus$ denotes the concatenation and t_i denotes the last item in $\mathbf{s}_i$. Although all attributed sequences in the dataset are zero-padded to the maximum length $t_{\max}$, the padded zero values are masked and not used in the computation. We model the process of predicting the label for each attributed sequence as:

$$\Theta(p_i) = \begin{cases} \sigma(\mathbf{W}_{\mathrm{p}}\boldsymbol{p}_i + \mathbf{b}_{\mathrm{p}}), & \text{ASA or No Attention} \\ \sigma\left(\mathbf{W}_{\mathrm{p}}\boldsymbol{\alpha}_i^{(t_i)} + \mathbf{b}_{\mathrm{p}}\right), & \text{ASHA} \end{cases}$$

where σ is a `sigmoid` activation function and $\hat{l}_i = \Theta(p_i)$ is the predicted label. The $\mathbf{W}_{\mathrm{p}}$ and $\mathbf{b}_{\mathrm{p}}$ are both trainable in our model.

3.3 Training.

3.3.1 Regularization. We adopt multiple strategies for different components in our `AMAS` network. We empirically select the following regularization strategies in our model based on: (1). For `Sequence Net`, we apply ℓ_2-regularization to the recurrent unit. (2).

Table 2: Compared Methods

Name	Data Used	Attention	Note
BLA	Attributes	No	[12]
BLS	Sequences	No	[28]
BLAS	Attributes Sequences	No	[35]
SOA	Sequences	Yes	[34]
ASA	Attributes Sequences	Yes	This paper
ASHA	Attributes Sequences	Yes	This paper

Dropout with a rate of 0.5 is used to regularize the fully connected layer in `Attribute Net`. (3). Lastly, we use Dropout with a rate of 0.2 in other fully connected layers in the model. Based on our observations, using regularization on `Attention Block` has no significant impact on the performance of `AMAS`.

3.3.2 Optimizer. We use an optimizer that computes the adaptive learning rates for every parameters, referred to as Adaptive Moment Estimation (`Adam`) [15]. The core idea is to keep (1). an exponentially decaying average of gradients in the past and (2). a squared past gradients. `Adam` counteracts the biases as:

$$\widehat{\omega^{(t)}} = \frac{\beta_1 \omega^{(t-1)} + (1-\beta_1) m^{(t)}}{1-\beta_1^t}$$

$$\widehat{\nu^{(t)}} = \frac{\beta_2 \nu^{(t-1)} + (1-\beta_2) \left(m^{(t)}\right)^2}{1-\beta_2^t}$$

where β_1 and β_2 are the decay rates, and $m^{(t)}$ is the gradient. We adopt $\beta_1 = 0.9$ and $\beta_2 = 0.999$ as in [15]. Finally, the `Adam` updates the parameters as:

$$\gamma^{(t+1)} = \gamma^{(t)} - \frac{\rho}{\sqrt{\widehat{\nu^{(t)}}} + \epsilon} \widehat{\omega^{(t)}}$$

where ρ is a static learning rate and ϵ is a constant with a small value to avoid division errors, such as division by zero. We empirically select $\rho = 0.01$.

4 Experiments

4.1 Datasets. Our solution has been motivated by use case scenarios observed at Amadeus corporation. For this reason, we work with the log files of an Amadeus [2] internal application. The log files contain user sessions in the form of attributed sequences. In addition, we apply our methodology to real-world, publicly available Wikispeedia data [29] and Reddit data [16]. For each type of data, we sample two subsets and conduct experiments independently. We summarize the data descriptions as follows:

- **Amadeus data (AMS-1, AMS-2)**[1]. We sampled six datasets from the log files of an internal application at Amadeus IT Group. Each attributed sequence is composed of a user profile containing information (*e.g.,* system configuration, office name) and a sequence of function names invoked by web click activities (*e.g.,* login, search) ordered by time.
- **Wikispeedia data (Wiki-1, Wiki-2)**. Wikispeedia is an online game requiring participants to click through from a given start page to an end page using fewest clicks [29]. We select *finished* paths and extract several properties of each path (*e.g.,* , the category of the start path, time spent per click). We also sample six datasets from Wikispeedia. The Wikispeedia data is available through the Stanford Network Analysis Project[2] [17].
- **Reddit data (Reddit-1, Reddit-2)**. Reddit is an online forum. Two datasets that contain the content of reddit submissions are used. The Reddit data is available through the Stanford Network Analysis Project[3].

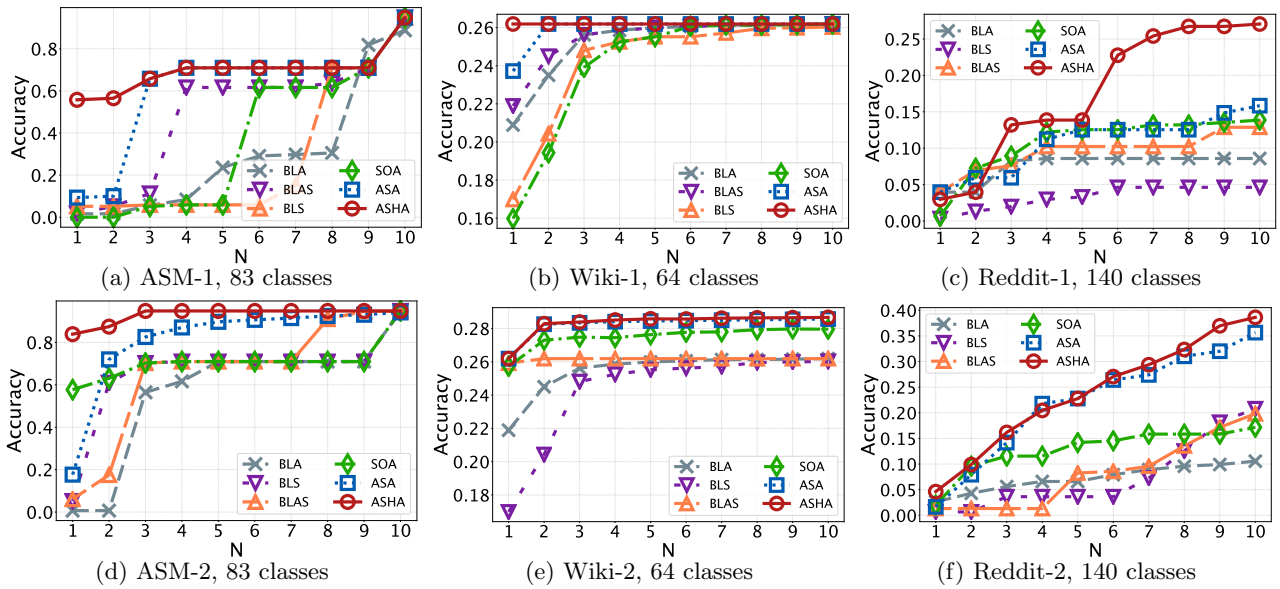
We use 60% of the instances in each dataset for the training and the rest 40% for testing. In the training, we holdout 20% of the training instances for validation.

4.2 Compared Methods. We evaluate our two approaches, namely ASA and ASHA and compare them with the following baseline methods. We summarize all compared methods used in this research in Table 2.

- BLA is built using a fully connected neural network to reduce the dimensionality of the input data, and then classify each instance.
- BLS classifies sequences only data using an LSTM.
- BLAS utilizes the information from both attributes and sequences. The resulting embeddings generated by BLAS are then used for classification.
- SOA builds attention on the sequence data for classification, while the attribute data is not used.

4.3 Experimental Setting. Our paper focuses on multi-class classification problem. We thus use accuracy as the metric to evaluate the performance. A higher accuracy score depicts more correct predictions of class labels. For each method, we holdout 20% as the validation dataset randomly selected from training dataset.

[1] Personal information is not collected.
[2] https://snap.stanford.edu/data/wikispeedia.html
[3] https://snap.stanford.edu/data/web-Reddit.html



(a) ASM-1, 83 classes (b) Wiki-1, 64 classes (c) Reddit-1, 140 classes

(d) ASM-2, 83 classes (e) Wiki-2, 64 classes (f) Reddit-2, 140 classes

Figure 4: Performance comparison on all six datasets.

For each experimental setting, we report the top-1 $\sim$ top-10 accuracy for each method.

We initialize our network using the following strategies: orthogonal matrices are used to initialize the recurrent weights, normalized random distribution [9] is used to initialize weight matrices in `Attribute Net`, and bias vectors are initialized as zero vector $\mathbf{0}$.

4.4 Accuracy Results. In Figure 4, we compare the performance of our ASA and ASHA solutions with the other state-of-the-art methods in Table 2. ASHA achieves the best performance of top-1 accuracy on most datasets. In most cases, ASHA outperforms other solutions significantly. We also observe a significant performance improvement by ASA compared to other methods. That is, although the top-1 accuracy performance of ASA is beneath that of ASHA, it still outperforms SOA with sequence-only attention and all other methods without attention. The two closest competitors, the SOA utilizing the attention mechanism and classifying each instance based on only the sequential data, and BLAS using information from both attributes and sequences, but without the help from an attention mechanism, are outperformed by the our proposed models.

4.5 Parameter Sensitivity Analysis

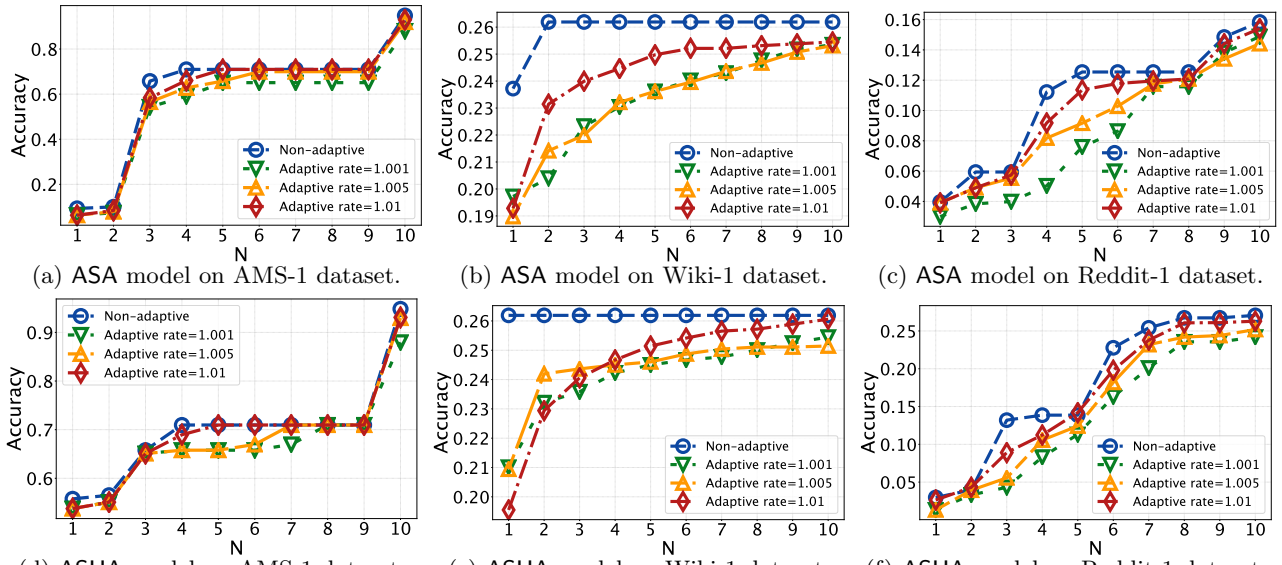
4.5.1 Adaptive Sampling Accuracy. As pointed out in recent work [4], adaptive sampling is capable of improving the efficiency of the optimization processes by adapting the training sample size in each iteration (*i.e.,* epoch). In this set of experiments, we evaluate the two models with varying adaptive sampling rates. We adopt the adaptive sampling function as:

$$N_\tau = N_1 \lambda^{(\tau-1)}$$

Here, τ denotes the epoch number, N_τ denotes the number of instances used in the τ-th epoch and λ the rate of adaptive sampling. We choose $\lambda = 1, 1.001, 1.005$, and 1.01 in our experiments, where $\lambda = 1$ means no adaptive sampling. The results presented in Figure 5 shows that the adaptive sampling with the above sampling rates can achieve similar performance as the non-adaptive approach yet now with much less training data.

4.5.2 Training with Adaptive Sampling. With the continuously increasing amount of training instances, we expect the history of training loss to be "jittery" when a model encounters previously unseen new instances. Different from previous experiments, where we use `Early Stopping` strategy to avoid overfitting, we now set a fixed number of 144 epochs for the ASHA model and 97 epochs for ASA model and collect the history of training and validation to study the adaptive training strategy. In Figure 6a, we observe the training with adaptive sampling is more aggressive compared to the non-adaptive approach. From Figure 6b we conclude that with a higher adaptive rate, the model more easily becomes overfitted. Similar conclusion can also be made from Figure 6d. Selecting a higher adaptive sampling rate can shorten the training time but risking a higher chance of overfitting.

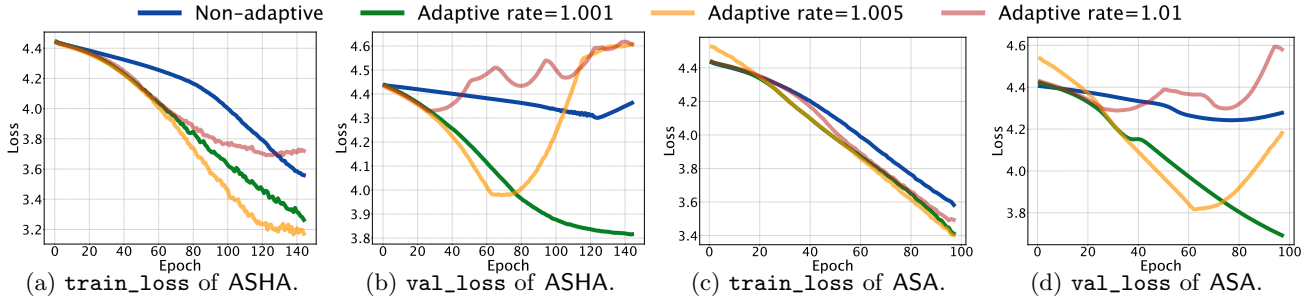
4.6 Case Studies. Figure 7 demonstrates the weights of each word of ten instances from the Reddit-2 dataset. Higher weights are represented with a darker



(a) ASA model on AMS-1 dataset. (b) ASA model on Wiki-1 dataset. (c) ASA model on Reddit-1 dataset.

(d) ASHA model on AMS-1 dataset. (e) ASHA model on Wiki-1 dataset. (f) ASHA model on Reddit-1 dataset.

Figure 5: The performance comparison between non-adaptive training and adaptive sampling.



(a) `train_loss` of ASHA. (b) `val_loss` of ASHA. (c) `train_loss` of ASA. (d) `val_loss` of ASA.

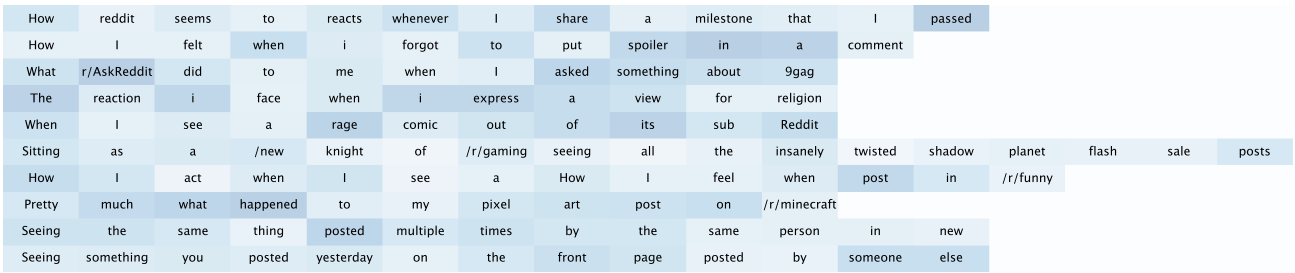
Figure 6: Comparison of the history of training and validation losses.

color, while lower weights are represented with a lighter color. Comparing the three cases, we find that the SOA has the most polarized weights among the three cases. This may be caused by the fact that the attention produced by SOA is solely based on the sequences, while ASHA and ASA have been influenced by attribute data.

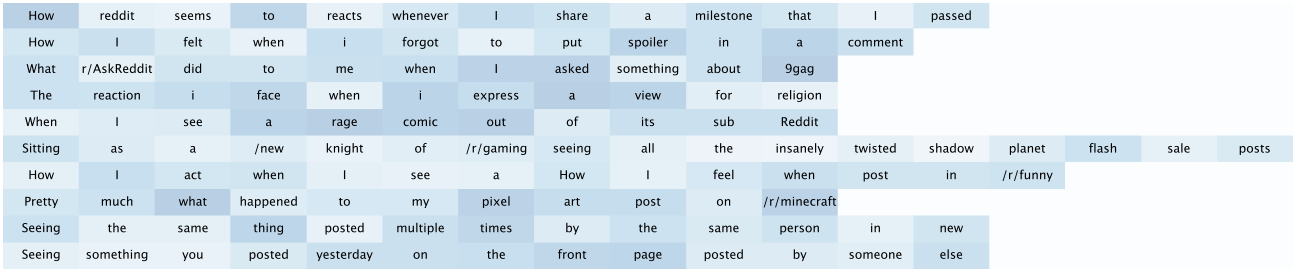
5 Related Work

5.1 Deep Learning Deep learning has received significant interests in recent years. Various deep learning models and optimization techniques have been proposed in a wide range of applications such as image recognition [14, 32] and sequence learning [5, 24, 33, 22]. Many of these applications involve the learning of a single data type [5, 24, 33, 22], while some applications involve more than one data type [14, 32]. The application of deep learning in sequence learning is popular with one of the most popular works, sequence-to-sequence [24], using a long short-term memory model in machine translation. The hidden representations of sentences in the source language are transferred to a decoder to reconstruct in the target language. The idea is that the hidden representation can be used as a compact representation to transfer similarities between two sequences. Multi-task learning [18] examines three multi-task learning settings for sequence-to-sequence models that aim to share either an encoder or a decoder in an encoder-decoder model setting. Although the above work is capable of learning the dependencies within a sequence, none of them focuses on learning the dependencies between attributes and sequences. Multimodal deep neural networks [14, 23, 32] are designed for information sharing across multiple neural networks, but none of these works focuses on the attributed sequence classification problem we target here.

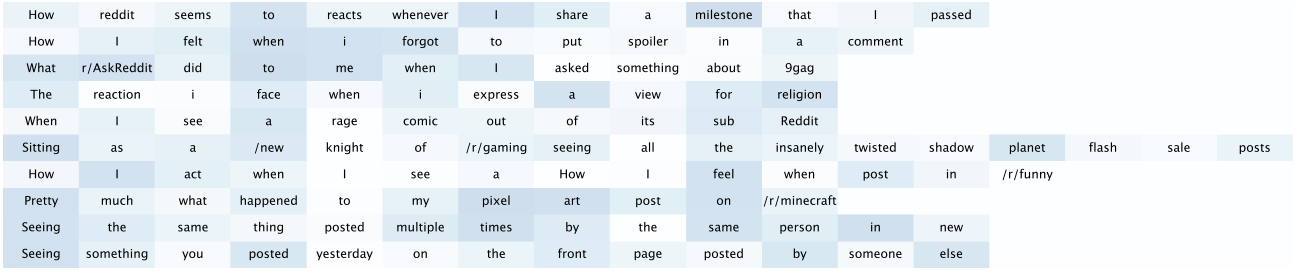
5.2 Attention Network. Attention network [20] has gained a lot of research interest recently. The attention network has been applied in various tasks, including image captioning [32, 21], image generation [11], speech recognition [6] and document classification [34]. The goal of using an attention network in these tasks is to make the neural network focus on the "interesting"



(a) ASHA



(b) ASA



(c) SOA

Figure 7: Weights of each words from 10 instances in Reddit-2. Higher weights are darker.

parts of each input, such as, a small region of an image or words that are most helpful to classifying documents. There are variations of attention networks, including *hierarchical attention* [34] and *dual attention* [21].

5.3 Sequence Mining. Recent work in sequence mining area aims to find the most frequent subsequence patterns [19, 8]. Several recent works [3, 19] focus on finding the most frequent subsequence that meets certain constraints. That is, they find the set of sequential patterns satisfying various linguistic constraints (e.g., syntactic, symbolic). Many sequence mining works focus on frequent sequence pattern mining. Recent work in [19] targets finding subsequences of possible non-consecutive actions constrained by a gap within sequences. [7] aims at solving pattern-based sequence classification problems using a parameter-free algorithm from the model space. [8] builds a subsequence interleaving model for mining the most relevant sequential patterns. However, none of the above works supports attribute data alongside the sequences, nor do they classify attributed sequences.

6 Conclusion

In this paper, we propose a AMAS framework with two models for classifying attributed sequences. Our ASHA and ASA models progressively integrate the information from both attributes and sequences while weighting each item in the sequence to improve the classification accuracy. Experimental results demonstrate that our models significantly outperform state-of-the-art methods.

7 Acknowledgement

We would like to thank the Amadeus IT Group for the generous supports in funding, computing resources and data access.

References

[1] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid, *Label-embedding for attribute-based classification*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013, pp. 819–826.

[2] Amadeus, *Amadeus IT Group*. http://www.amadeus.com. Accessed: 2017-09-23.

[3] N. Béchet, P. Cellier, T. Charnois, and B. Crémilleux, *Sequence mining under multiple constraints*, in SIGSAC, 2015, pp. 908–914.

[4] J. Chen, S. Sathe, C. Aggarwal, and D. Turaga, *Outlier detection with autoencoder ensembles*, in SDM, SIAM, 2017, pp. 90–98.

[5] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, *Learning phrase representations using rnn encoder-decoder for statistical machine translation*, arXiv preprint arXiv:1406.1078, (2014).

[6] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, *Attention-based models for speech recognition*, in NIPS, 2015, pp. 577–585.

[7] E. Egho, D. Gay, M. Boullé, N. Voisine, and F. Clérot, *A parameter-free approach for mining robust sequential classification rules*, in ICDM, 2015, pp. 745–750.

[8] J. Fowkes and C. Sutton, *A subsequence interleaving model for sequential pattern mining*, arXiv preprint arXiv:1602.05012, (2016).

[9] X. Glorot and Y. Bengio, *Understanding the difficulty of training deep feedforward neural networks*, in International Conference on Artificial Intelligence and Statistics, 2010, pp. 249–256.

[10] A. Graves, *Generating sequences with recurrent neural networks*, arXiv preprint arXiv:1308.0850, (2013).

[11] K. Gregor, I. Danihelka, A. Graves, D. Rezende, and D. Wierstra, *Draw: A recurrent neural network for image generation*, in ICML, 2015, pp. 1462–1471.

[12] G. E. Hinton and R. R. Salakhutdinov, *Reducing the dimensionality of data with neural networks*, science, 313 (2006), pp. 504–507.

[13] S. Hochreiter and J. Schmidhuber, *Long short-term memory*, Neural computation, (1997).

[14] A. Karpathy and L. Fei-Fei, *Deep visual-semantic alignments for generating image descriptions*, in CVPR, 2015, pp. 3128–3137.

[15] D. P. Kingma and J. L. Ba, *Adam: Amethod for stochastic optimization*.

[16] H. Lakkaraju, J. J. McAuley, and J. Leskovec, *What's in a name? understanding the interplay between titles, content, and communities in social media.*, ICWSM, 1 (2013), p. 3.

[17] J. Leskovec, *Wikispeedia navigation paths*. `https://snap.stanford.edu/data/wikispeedia.html`. Accessed: 2018-04-09.

[18] M.-T. Luong, Q. V. Le, I. Sutskever, O. Vinyals, and L. Kaiser, *Multi-task sequence to sequence learning*, arXiv preprint arXiv:1511.06114, (2015).

[19] I. Miliaraki, K. Berberich, R. Gemulla, and S. Zoupanos, *Mind the gap: Large-scale frequent sequence mining*, in SIGMOD, 2013, pp. 797–808.

[20] V. Mnih, N. Heess, A. Graves, et al., *Recurrent models of visual attention*, in NIPS, 2014.

[21] H. Nam, J.-W. Ha, and J. Kim, *Dual attention networks for multimodal reasoning and matching*, (2017).

[22] P. Neculoiu, M. Versteegh, M. Rotaru, and T. B. Amsterdam, *Learning text similarity with siamese recurrent networks*, Proceedings of the 1st Workshop on Representation Learning for NLP, (2016), p. 148.

[23] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, *Multimodal deep learning*, in ICML, 2011, pp. 689–696.

[24] I. Sutskever, O. Vinyals, and Q. V. Le, *Sequence to sequence learning with neural networks*, in NIPS, 2014, pp. 3104–3112.

[25] A. Tajer, V. V. Veeravalli, and H. V. Poor, *Outlying sequence detection in large data sets: A data-driven approach*, IEEE Signal Processing Magazine, 31 (2014), pp. 44–56.

[26] J. Wan, D. Wang, S. C. H. Hoi, P. Wu, J. Zhu, Y. Zhang, and J. Li, *Deep learning for content-based image retrieval: A comprehensive study*, in Proceedings of the 22nd ACM international conference on Multimedia, ACM, 2014, pp. 157–166.

[27] W. Wang, Y. Huang, Y. Wang, and L. Wang, *Generalized autoencoder: A neural network framework for dimensionality reduction*, in CVPR, 2014, pp. 490–497.

[28] Y. Wang and F. Tian, *Recurrent residual learning for sequence classification*, in EMNLP, 2016, pp. 938–943.

[29] R. West, J. Pineau, and D. Precup, *Wikispeedia: An online game for inferring semantic distances between concepts*, in IJCAI, 2009.

[30] E. P. Xing, M. I. Jordan, S. J. Russell, and A. Y. Ng, *Distance metric learning with application to clustering with side-information*, in NIPS, 2003, pp. 521–528.

[31] Z. Xing, J. Pei, and E. Keogh, *A brief survey on sequence classification*, SIGKDD, 12 (2010), pp. 40–48.

[32] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, *Show, attend and tell: Neural image caption generation with visual attention*, in ICML, 2015, pp. 2048–2057.

[33] Y. Xu, J. H. Lau, T. Baldwin, and T. Cohn, *Decoupling encoder and decoder networks for abstractive document summarization*, MultiLing 2017, (2017), p. 7.

[34] Z. Yang, D. Yang, C. Dyer, X. He, A. J. Smola, and E. H. Hovy, *Hierarchical attention networks for document classification*, in HLT-NAACL, 2016, pp. 1480–1489.

[35] Z. Zhuang, X. Kong, E. Rundensteiner, J. Zouaoui, and A. Arora, *Unsupervised embedding of attributed sequences: A multimodal learning approach*, in submission, 2018.