Semi-Supervised Confidence Network aided Gated Attention based Recurrent Neural Network for Clickbait Detection

Amrith Rajagopal Setlur
amrithsetlur22@gmail.com

Abstract

Clickbaits are catchy headlines that are frequently used by social media outlets in order to allure its viewers into clicking them and thus leading them to dubious content. Such venal schemes thrive on exploiting the curiosity of naive social media users, directing traffic to web pages that won't be visited otherwise. In this paper, we propose a novel, semi-supervised classification based approach, that employs attentions sampled from a Gumbel-Softmax distribution to distill contexts that are fairly important in clickbait detection. An additional loss over the attention weights is used to encode prior knowledge. Furthermore, we propose a confidence network that enables learning over weak labels and improves robustness to noisy labels. We show that with merely 30% of strongly labeled samples we can achieve over 97% of the accuracy, of current state of the art methods in clickbait detection.

1 Introduction

With the number of social media users increasing by the day, one of the prime objectives of online news media agencies is to lead social media users onto bogus pages through the display of luscious text/images (Loewenstein, 1994). In most cases the content on the landing page is disparate to the headline the user clicked on. Source verification is no longer warranted as news agencies aren't held accountable for the content they post online. As (M. Potthast and Hagen, 2016) suggests, at least 15 of the most prominent content creators use clickbaits in some form or the other to honey-trap users. Impetus for such schemes can range from directing traffic to web sites that force users to purchase a product, to shaping popular opinion especially during elections. Some clickbaits claim to accomplish inconceivable tasks while others rely on a viewer's inducement to grapevines.

- "You will never believe what this celebrity did at the awards ceremony."
- "10 things that will get you fairer in 5 days."
- "Millionaires want to conceal this scheme. It can make you rich in a week."

Earlier approaches on tackling clickbaits mainly focused on cheap and easy to implement solutions. Blacklisting URLs has been, to some extent, effective in regulating an average user's exposure to clickbaits. (Gianotto, 2014) assumed that most clickbaits are based on a few key phrases, and a naive way yet effective strategy would be to simply look for such phrases. Such an assumption holds no ground today. As the problem grew to be more pervasive, social media companies modeled the probability for a content to be a clickbait based on factors like click-to-share ratio, time spent by user on the target page, and other such quantifiers. Recent research focuses on salvaging sentence structures, n-grams & embeddings among other features, in classifiers like Random Forests (RF), Gradient Boosted Trees (GBT), Support Vector Machines (SVM) or the vanilla Logistic Regression (LR). With the advent of online news agencies, there exists a plethora of such sources, but labeling each of the headlines from them would be a staggering task. This vindicates the need for an unsupervised / semi-supervised approach.

Contributions of this paper include: 1. A novel loss component on the attention weights that encodes prior information from a weak source of labels, which eventually improves the generalizability of the deep learning model that has been trained on a small representative dataset. 2. A joint architecture that incorporates into the clickbait classification framework a confidence network that tackles label noise. 3. Using Gumbel-Softmax for gated attentions, thus enforcing peaky attentions over word contexts. 4. Empirically proving the performance of the proposed approach on popular clickbait datasets with only a small portion of labeled samples.

2 Related Work

The problem of clickbait detection has been primarily studied in two forms. One of them involves a pair of post and target phrases, in which, the objective is to identify whether the post text (visible to the user), is in someway related to the target content (text/images on the landing page). Using this formulation, (P. Biyani and Blackmer, 2016) suggested that the features involved in clickbait detection can be broadly classified into: content features (quotes, capitalization), similarity between the source and target representations, informality, forward reference, URLs etc. A gradient boosted tree was trained using these features. (Chakraborty et al., 2016) highlighted the use of features based on linguistic and structural differences between the clickbait & non-clickbait headlines. Using the n-grams and word patterns, they successfully trained a SVM classifier with a Radial Basis Function (RBF) kernel, that outperformed the baseline rule based method in (Giannotto, 2014).

The other form comprises of making decisions purely based on the headline content (Zhou, 2017), (P. Biyani and Blackmer, 2016). Our work is based on this paradigm, and performs on par with methods that follow the former approach. According to (Schuster and Paliwal, 1997), Recurrent Neural Network (RNN) based sentence embeddings of the headlines, are expressive enough to learn neural network based classifiers that separate the two classes. (Teng et al., 2017) explored convolutional networks that can convolve our word embeddings to learn n-grams, sub-words and token patterns that are consistent with clickbaits. (M. Potthast and Stein, 2017) worked on the twitter dataset and treated clickbait detection as a regression problem instead of binary classification. Hence, they proposed a model that outputs scores indicative of how clickbaity a tweet is. (A. Anand and Park, 2017) used a bi-directional LSTM to improve upon the results published by (Chakraborty et al., 2016), on the *Headlines Dataset* (Section 3.1).

(Wang et al., 2016) employed attention based mechanisms (Bahdanau et al., 2014) for text classification. Our work reuses the self-attentive structured sentence embeddings introduced by (Z. Lin and Bengio, 2017), with some additional components that supplant text classification in semi-supervised environments, with weakly labeled data.

Some of the most recent approaches that have been successful in modelling curiosity are based on the ”novelty” and ”surprise” components of a headline. (Venneti and Alam, 2018) used topic modelling to identify the topic distributions for each headline in the corpus. Distance metrics in the space of propability distributions like KL divergence and Hellinger distance were used as features to express novelty while surprise was primarily modeled using bi-gram frequency counts. Based on these features, an SVM/LR classifier was trained on a small section of the training data.

3 Proposed Methodology

3.1 Dataset

The first dataset used in the paper is the *Headlines Dataset* curated by (Chakraborty et al., 2016) [1] and used by (A. Anand and Park, 2017), (Rony et al., 2017) et al. It holds labels for 32,000 headlines which featured in articles from BuzzFeed, New York Times, Scoopwhoop, Upworthy, The Guardian, ViralNova, The Hindu, ViralStories, Thatscoop and WikiNews. In all, there are 15,999 clickbait and 16,001 non-clickbait samples. We perform an ablation study using a varying proportion of the dataset as our strongly labeled set. Furthermore, we compare our results against strong baselines established on this dataset, in the absence of labeled data. All experiments have been performed using a 5-fold cross validation scheme, in order to reconcile with the existing baselines.

The second dataset was picked from the *Clickbait-Challenge 2017* (Zhou, 2017)[2]. The challenge posed clickbait detection as a regression problem, assigning each entry a set of scores from five different annotators. Each $score \in$

[1]https://github.com/bhargaviparanjape/clickbait/

[2]https://www.clickbait-challenge.org/

$\{0, \frac{1}{3}, \frac{2}{3}, 1\}$[3]. Therefore, each sample was annotated with score summary statistics: truthMean, truthJudgements, truthMode, and truthMedian. For the sake of consistency we reformulated this as a classification problem using the following decision:

$$C_i = \begin{cases} clickbait & \text{if } truthMean_i \geq 0.5 \\ non-clickbait & otherwise \end{cases} \quad (1)$$

Although the dataset consists of "targetText" and "image" (landing page) data apart from "postText" (headline), we were able to attain convincing results by using features derived purely from "postText". This assumption was based on the behaviour of an average annotator, as in most cases the annotator judgments are purely hinged on the headline. Based on similar assumptions, (A. Anand and Park, 2017), (Zhou, 2017) and (Rony et al., 2017), used n-grams, simple word filters or latent text representations (LSTMs) that were solely based on headline content. Following 3 splits were provided in the Clickbait-Challenge (C: Clickbaits, NC: Non-clickbaits).

Label	Headlines	C	NC
A	19,538	4,761	14,777
B	2,495	762	1,697
C	80,012	-	-

We evaluated F1-score and accuracy on the test set B while using portions of the set A as our labeled dataset (with the remaining as unlabeled), bootstrapped with the unlabeled set C.

3.2 Random Forest

In order to identify words of high importance (information gain), we trained a Random Forest Classifier on the strongly labeled section of the dataset. By using entropy (Eq. 2) as a measure of information gain, while splitting samples at each node of a tree, we posited that words with low entropy (summarized in Table 1) were strong signals for identifying clickbaits. Before fitting the random forest, the headlines were pre-processed; all numeric content was mapped to <n-token>, URLs were mapped to <u-token>. Along with these, entity detectors were useful in identifying references to dates/years, locations. The Wordnet Lemmatizer was used to obviate trivial variances in word representations. The analyses presented in Table 1 was done on the *Headlines Dataset*. In this section, we propose the use of Random Forest as a source of weak labels, with the Random Forest being trained on the human labeled samples.

[3]https://www.clickbait-challenge.org/#data

Word	Importance	Naive Inclination
<n-token>	5.120	clickbait
like	3.112	clickbait
dies	3.042	non-clickbait
people	2.351	clickbait
know	2.336	clickbait
life	2.155	clickbait
need	2.062	clickbait
president	1.799	non-clickbait
wins	1.664	non-clickbait
kill	1.623	non-clickbait
iraq	1.244	non-clickbait
hilarious	1.058	clickbait
favorite	1.039	clickbait
laugh	0.965	clickbait
really	0.963	clickbait
court	0.941	non-clickbait
china	0.940	non-clickbait
dead	0.891	non-clickbait
photos	0.869	clickbait
most	0.851	clickbait
leader	0.702	clickbait
pictures	0.698	clickbait
obama	0.592	non-clickbait
questions	0.524	clickbait

Table 1: Some of the tokens with high word-importance factor (Inverse Entropy).

$$\begin{aligned} E_s &= \frac{N_l}{N} E_l + \frac{N_r}{N} E_r \\ E_l &= -\sum_{i \in C} p_{i_l} \log p_{i_l} \\ E_r &= -\sum_{i \in C} p_{i_r} \log p_{i_r} \end{aligned} \quad (2)$$

p_{i_l}:	proportion of samples of the left split that $\in$ class C_i
p_{i_r}:	proportion of samples of the right split that $\in$ class C_i
N_l:	number of samples in left split
N_r:	number of samples in right split
N:	total number of samples

Salvaging the tokens with the lowest entropy, we built simple rules to detect clickbaits. Tree paths that included decisions based on these low entropy tokens were used to construct rules in Disjunctive Normal Forms (DNFs) (Table 2). The unlabeled dataset was then run through these rules to determine weak labels for classification. This aided the training of the attention network, as corroborated by our experiments (Section 4).

3.3 Problem Definition

We are given two sets of data, namely: $D_s = \{(X_1, y_1), (X_2, y_2), .., (X_n, y_n)\}$, that is strongly labeled through manual annotations and $D_w = \{(X_1, \hat{y}_1), (X_2, \hat{y}_2), .., (X_n, \hat{y}_n)\}$, which is weakly labeled and is composed of samples from the unlabeled set. The weak labels in D_w are generated by the Random Forest Classifier (RF) (Section 3.2). In addition to this, we assimilate $\hat{D}_s$, that is composed of RF predictions on D_s (Eq. 4).

$$\begin{gathered} \hat{y}_i = RF(X_i) \\ \forall (X_i, \hat{y}_i) \in D_w \end{gathered} \tag{3}$$

$$\begin{gathered} \hat{D}_s = \{(X_i, y_i, RF(X_i))\} \\ \forall (X_i, y_i) \in D_s \end{gathered} \tag{4}$$

The goal would be to train a classification network on the set obtained by concatenating D_s & D_w. It is assumed that D_s is comprised of a representative set of samples and that D_s and D_w contain i.i.d. samples from the true data distribution. Since D_w consists of weak labels from the Random Forest and $|D_w| > |D_s|$, we propose the use of a confidence network (Section 3.7) that predicts the accuracy of a weak label. These accuracy scores would re-weigh the gradient updates when the loss is calculated using samples from D_w, thus attenuating the effect of noisy labels.

3.4 Deep Attention Network

Self-attentive structured attention mechanisms for efficient semantic latent sentence representations was proposed by (Z. Yang and Hovy, 2016) and (Z. Lin and Bengio, 2017). (Zhou, 2017) further ascertained the effectiveness of the attention mechanism in clickbait detection. The intuition behind a self attention mechanism is that the network learns the importance of each token's context, for the task in hand (clickbait detection), along with a hidden state representation of the word context itself. While (Zhou, 2017), (Z. Yang and Hovy, 2016) used this in a fully supervised setting, we researched its resilience under the semi-supervised tone. In the following sections, we show that, the presence of an external attention enforcer is pivotal in training the network. Merely training a deep network on a meagre set of a few thousand samples leads to overfitting (Goodfellow et al., 2016), and as expected, this claim was substantiated, when we noted a higher validation/test loss, upon direct application of the architecture proposed by (Zhou, 2017), (Z. Lin and Bengio, 2017).

We propose vital and consequential modifications to the base network which accommodates the semi-supervised learning problem. Attention mechanisms have traditionally been utilized to focus attention on features that are most influential in performing a specific task, like image captioning or semantic segmentation, in the image domain (Xu et al., 2015). Drawing a parallel, we propose an attention based loss that forces the attentions for a specific set of words to be higher than the rest. When a large number of samples are available, the attention module is self-sufficient in determining such tokens. In this case however, we use as a surrogate, the word importance measures from the Random Forest classifier to identify them (Section 3.6).

3.5 Architecture

Figure 1 demonstrates the two networks employed during training. The classification/self-attention network determines the sentence embedding post a dot-product operation on the LSTM hidden state representations using the attention weights. Given a headline with N tokens, we map each token w_i, where $i \in \{1, \ldots, N\}$, to its corresponding glove embedding[4] (trained on the twitter corpus), denoted by x_i. A bi-directional LSTM network, encodes the word context (from both directions), for the word w_i, in the time step t_i (Eqs. 5).

$$\begin{gathered} h_{left_i} = LSTM_{left}(x_i) \\ h_{right_i} = LSTM_{right}(x_i) \\ h_i = h_{left_i} \,||\, h_{right_i} \\ x_i \in R^e \\ h_{left_i}, h_{left_i} \in R^u \\ h_i \in R^{2u} \end{gathered} \tag{5}$$

Given the word context embeddings, the at-

[4]https://nlp.stanford.edu/projects/glove/

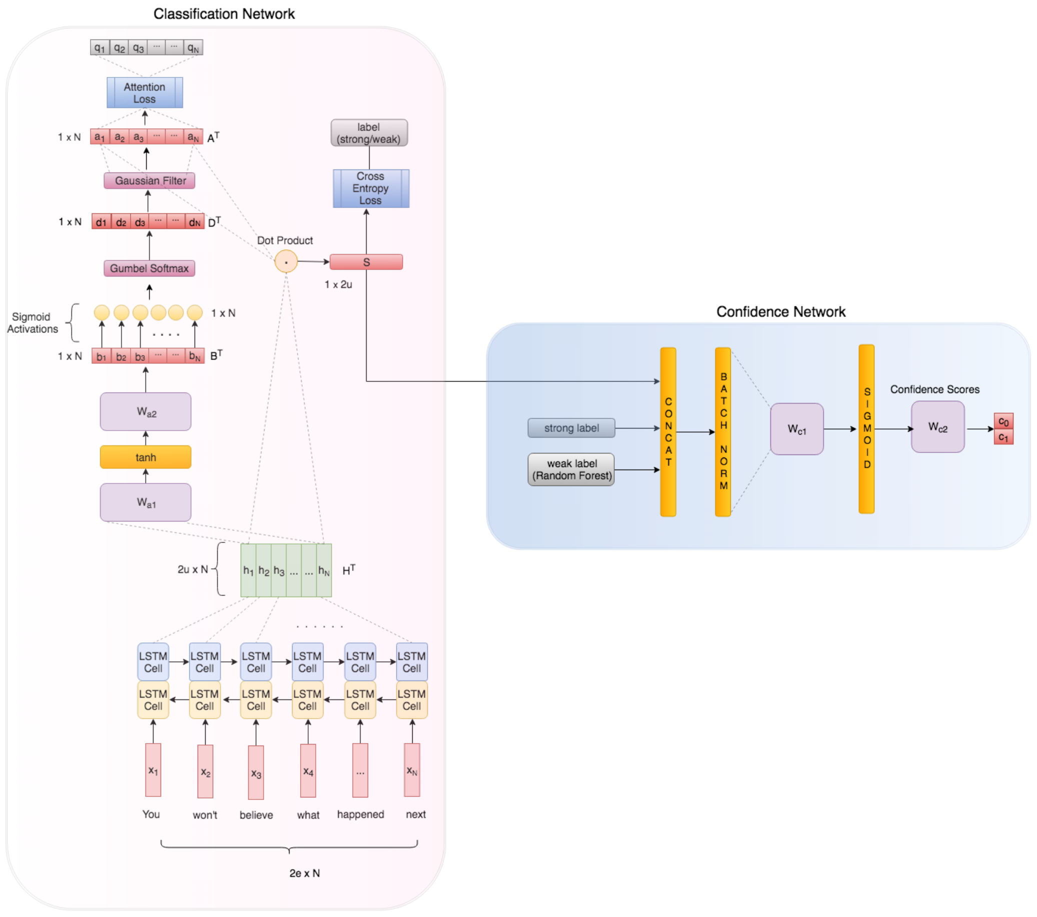

Figure 1: Network Architecture

tention network maps them to intermediate token level activations, using a Multi-Layered Perceptron (MLP) network with non-linear (tanh) activations. With $H \in R^{N\times 2u}$ as the context embedding matrix, the network outputs $B \in R^N$, as the intermediate activation vector (Eq. 6).

$$B = \sigma((\tanh(H \cdot {W_{a1}}^T)) \cdot W_{a2}) \\ H \in R^{N\times 2u}, W_{a1} \in R^{m\times 2u}, W_{a2} \in R^{m\times 1} \quad (6)$$

The original structured self-attentive network (Z. Lin and Bengio, 2017) proposed a softmax activation in the final layer. This was befitting in the case of sentiment classification, where in the sentiment of each sentence was pivoted on a few key tokens and such tokens existed in all sentences, irrespective of the class. On the contrary, in the case of clickbait detection, it is mostly the negative class (clickbaits), that contains such tokens of importance. As depicted in Table 1, the words identified by the Random Forest for the clickbait class are independent of the news item/news subject (“<n-token>”, “know”, “need”, “favorite”, “most”). Same can’t be said for the positive class (“obama”, “iraq”, “china”, “president”). Hence we introduced a sigmoid layer to better suit the case of clickbait detection.

The intermediate activations $B \in R^N$, obtained post the sigmoid layer were treated as parameters of a binary distribution. Treating each b_i as $P(a_i = 1)$, we could sample from the binary distribution. To propagate losses during back-propagation, we instead sampled values from the Gumbel-Softmax distribution (Jang et al., 2016), to obtain $D \in R^N$ (Eq. 7). This encapsulates the “gated” portion of the network where in activations for word contexts are sampled from a discrete distribution whose parameters are learned. The central idea is to support a case where in the sentence may not have significant impact words (fairly common with the positive class), and with this formulation the sampled values can represent such a case trivially with a zero vector. In the original work done by (Z. Lin and Bengio, 2017), such activations were inconceivable.

$$d_i = \frac{\exp\left((\log(b_i) + g_{i_1})/\tau\right)}{\exp\left((\log(b_i) + g_{i_1})/\tau\right) + \exp\left((\log(1-b_i) + g_{i_0})/\tau\right)} \quad (7)$$

$g_{i_1} \sim$ Gumbel(0, 1)
$g_{i_0} \sim$ Gumbel(0, 1)
τ: the temperature parameter that determines the extent to which d_i would be close to 0 or 1

Adding the Gumbel-Softmax enables the possibility of peaky activations. Lower values of τ would lead to d_i lying on either end of the spectrum, i.e 0 or 1, thus resembling a sample from a binomial distribution.

In cases of very low volumes of labeled data we observed that the attention weights of tokens neighboring the words of importance were very close to 0, and thus their context representations wasn't contributing towards the final sentence representation. To avoid this, we used a Gaussian filter over the samples from the Gumbel-Softmax distribution, which acts as a 1D convolution. The weights of the Gaussian filter are not learned. They are fixed and normalized so as to ensure that the final attention weights $0.0 \leq a_i \leq 1.0$. The activation vector $A \in R^N$, and LSTM hidden states H produce the final sentence embedding S (Eq. 8).

$$\begin{gathered} A = GaussianFilter(B) \\ S = H^\top \cdot A \\ H \in R^{N \times 2u},\ A \in R^N,\ S \in R^{2u} \end{gathered} \quad (8)$$

3.6 Attention Loss

One way to identify key words/features is to use a weak labeler like Random Forest. It identifies a set of words crucial in decision making (Table 1). The loss penalizing a deviation from such important words, is the standard binary cross entropy loss where it considers the attentions sampled from the Gumbel-Softmax distribution. The true attention is inferred from the word importance or entropy scores generated by the Random Forest. A true attention of 1.0 is assigned to words with importance over a particular threshold.

The set of positive activations in the corpus is much lower than its counterpart. This class imbalance was tackled by simply using the class weights to re-weigh the attention loss (Alejo et al., 2017). The proportion of activations of class i is inversely proportional to w_i (Eq. 9). The parameters of the classification network in Figure 1 are optimized over a combination of the attention and classification losses (Eq. 10).

$$\begin{gathered} \mathcal{L}_a(X, y) = -W^\top [Q \odot log(A) + \bar{Q} \odot log(\bar{A})] \\ A = \{a_1, \ldots, a_N\}^\top,\ \bar{A} = 1 - A \\ Q = \{q_1, \ldots, q_N\}^\top,\ \bar{Q} = 1 - Q \\ W = \{w_1, \ldots, w_N\}^\top \end{gathered} \quad (9)$$

A:	network activations for tokens
Q:	true activations for tokens
W:	sample weights
N:	number of tokens
$\odot$:	Hadamard product
X, y:	samples $\sim D_s, D_w$

$$\mathcal{L}_{ca}(X, y) = \mathcal{L}_c(X, y) + \lambda \cdot \mathcal{L}_a(X, y) \quad (10)$$

X, y:	samples $\sim D_s, D_w$
$\mathcal{L}_{ca}$:	Classification + Attention Loss
$\mathcal{L}_c$:	Classification Loss
$\mathcal{L}_a$:	Attention Loss
λ:	Contribution of the attention loss to the total loss

3.7 Optimization Algorithm

Amalgamation of strong and weak supervision has been used to solve constraints like data paucity, noisy labels and a few others (Schapire, 1990). Weak learners are sources of noisy labels. (Dehghani et al., 2017b) proposed the use of a confidence network to estimate the accuracy of a noisy label. Similar to (Dehghani et al., 2017a) and (Arachie and Huang, 2018), we used an optimization method that is a variant of the standard Stochastic Gradient Descent (SGD). The latter uses a constant learning rate on all samples in an iteration. Such an approach can lead to noisy gradients, especially when the proportion of strongly labeled samples in a batch is low (Goodfellow et al., 2016).

In order to mitigate this, a confidence network, trained on $\hat{D}_s$, formulates the confidence in a weak label as a function of the weak label (Random Classifier output) and the encoded input representation. As shown in Figure 1, the architecture is split into the classification and confidence networks. Batches of i.i.d samples are drawn iteratively from $\hat{D}_s$ and D_w. A batch of the former type is used train the classification network, the weights for which are updated using SGD. Since the samples from $\hat{D}_s$ also consist of RF predictions on strongly labeled samples, they are used to train the confidence network (f_{conf}) as well.

$$\begin{aligned} score = f_{conf}(E(X), \hat{y}) \\ \forall (X, y, \hat{y}) \in \hat{D}_s, \\ E(X) : latent\, representation\, for\, X \end{aligned} \tag{11}$$

For a sample $(X, y, \hat{y}) \sim \hat{D}_s$, a forward propagation on the classification network would generate the sentence representation S (denoted as $E(X)$ in Eq. 11). S is concatenated with $\hat{y}$, and passed to a batch-normalization layer. Since the embeddings and binary signals lie on separate manifolds, the batch-normalization is quintessential before the concatenated input is passed to a neural network. Figure 1 enunciates the output of the network as a 2-dimensional vector $[c_0, c_1]$. The $score$ in Eq. 11 is given by c_1. The confidence network is trained using a cross-entropy loss ($\mathcal{L}_{conf}$ in Eq. 12). The true confidence is given by the indicator $\mathbb{1}_{y=\hat{y}}$.

In the subsequent iteration when samples are drawn from D_w confidence scores are inferred on each sample in this set through a forward propagation on the confidence network. These scores are passed to the optimization algorithm for the classification network in order for it to re-weigh its gradient updates. Equation 13 defines the gradient update rule for parameters in the classification network.

$$\mathcal{L}_{conf}(X, y, \hat{y}) = -\mathbb{1}_{y \neq \hat{y}} \log c_0 - \mathbb{1}_{y=\hat{y}} \log c_1 \tag{12}$$

$(X, y, \hat{y}) \sim \hat{D}_s$
$c : [c_0, c_1]$, output of the confidence network
$\mathcal{L}_{conf}$: Loss on sample $(X, y, \hat{y})$
$\mathbb{1}_{y=\hat{y}}$: True confidence value

$$\begin{aligned} \widehat{\nabla w} = \frac{1}{B} \sum_{i=1}^{B} f_{conf}(X_i, \hat{y}_i) \cdot \nabla_w \mathcal{L}_{ca}(X_i, \hat{y}_i) \\ w_{t+1} = w_t - \eta_t \widehat{\nabla w} \end{aligned} \tag{13}$$

η_t:	learning rate
B:	Batch size of samples drawn from $\hat{D}_s$
$\mathcal{L}_{ca}$:	Loss on sample $(X_i, \hat{y}_i)$ (Section 3.6)
f_{conf}:	Confidence network

Rule	Class
believe ∧ <n-token>	C
president ∧ iraq ∧ war	NC
hilarious ∧ photos ∧ <n-token>	C

Table 2: Rules drawn from the tress in RF.

4 Experiments

4.1 Experimental Setting

The following discussion on hyper-parameters is with respect to the *Headlines Dataset*. Although nearly the same set of values were applicable to the *Clickbait-Challenge Dataset*.

The hyper-parameters of the random forest were fine tuned using Bayesian optimization techniques. We used 50 estimators, with a maximum depth of 3 and a minimum split size of 5 along with the entropy criterion for splitting. Rules in DNF form were constructed by traversing paths that consisted only of tokens with high information gain. The threshold for minimum word importance was found to be optimal at 0.42. Table 2 lists some of the rules (mentioning only the words whose presence triggers the corresponding rule).

The dimensionality of the parameters entrenched in the classification and confidence networks have been encapsulated in Table 3. Glove embeddings (Pennington et al., 2014) trained on the twitter corpus, were used as base word embeddings, which were fed to the LSTM cells. Dropouts (Srivastava et al., 2014) have been commonly used in recurrent neural networks as a form of regularizer, especially after they were proven to be a form of Bayesian inference in deep Gaussian Processes (Gal and Ghahramani, 2016). We used a dropout parameter of 0.5 within the LSTM, and 0.4 for the inputs to the fully connected layer with weights W_{a2}. For the Gumbel-Softmax layer, temperature parameter of $\tau = 0.7$ was found to be optimal. Lower values of τ would have led to higher attention weights for the words of importance, but it would have also driven the weights for the rest of the tokens to zero. For the batch-normalization layer present within the confidence network, a momentum of 0.05 resulted in a slightly better validation accuracy as compared to the standard value of 0.1 (Ioffe and Szegedy, 2015). This can be attributed to the variance in label confidence across alternate mini-batches that were sampled randomly from the labeled and unlabeled sets. The

Parameter	Dimensionality
x (Word Embedding)	300
h (LSTM hidden state)	200
W_{a1}	32×400
W_{a2}	32×1
W_{c1}	64×400
W_{c2}[5]	65×2

Table 3: Dimensionality of the parameters in the classification and confidence networks.

standard mini-batch SGD optimizer with a learning rate of 0.0001 (Li et al., 2014) and a batch size of 64 samples was employed. The parameter λ, involved in the combination of attention and classification losses was fixed at 0.3. We used an early stopping criteria, to stunt training. In most cases, 5 epochs were sufficient to fit the training data, a result, which seems to corroborate with the size of the dataset involved.

4.2 Results

In accordance with the disparity of the datasets mentioned in section 3.1, we summarize the proposed model performance on the two of them independently. In both cases, we benchmark the model performance against baselines, that claim to have achieved state of the art results on the dataset in question.

(A. Anand and Park, 2017) claimed a high classification accuracy of 0.982 after having experimented with multiple RNN based architectures to embed the clickbait embeddings in a multi-dimensional space. Our model's performance emulates the former on the fully labeled dataset. On the partially labeled set we achieve a high accuracy of 0.980, even with only 30% of labeled samples (Table 4). This is an increment of **4.03%** in the accuracy on the validation set, when compared to the BiLSTM based network.

We further study model performances across the various model architectures supplanted in an incremental fashion (Table 5). When data paucity is high (30% labeled), we see significant differences in accuracy, precision and recall while adding individual components to the network. Increments have been noticed when a large number of labeled samples (80%) are available, albeit the rate of improvement is negligible. The gaussian filter over attention weights sampled from Gumbel-Softmax is more effective in the former case where scattering attention onto the neighborhood of high importance words prevents the sentence representation from collapsing to an average of word vectors in the inchoate stages of training. The precision and recall metrics also show similar trends, with increments of **3.92%** and **3.91%** respectively, in case of 30% labeled samples.

In case of the *Clickbait-Challenge* dataset (section 3.1), we train our model on the labeled split (A), concatenated with the unlabeled samples in (C) and observe model performance on the test set (B), consistent with (Zhou, 2017). We further performed experiments to study our model's ability to learn meaningful sentence representations when only a portion of the set A was labeled. Table 6 puts into perspective the observed Mean Squared Error (MSE), precision and accuracy on the test set, in comparison to the current best performing model (Zhou, 2017) on the dataset in question. The existing baseline involves sentence classification solely using the self-attention network introduced by (Z. Lin and Bengio, 2017). The baseline results are convincing when all labels are available and on par with our model's performance. On the other hand, when only 30% of the set A is used as the labeled set we observed a jump of **23.55%** & **25.61%** with regards to the accuracy & F1-score respectively.

Conclusion and Future Work

In this paper, we proposed a novel architecture to tackle clickbait detection when only a few labeled samples are present. We successfully showed that use of a weak labeler like Random Forest can generate priors over an attention mechanism and thus improve generalizability. Empirically, we have also shown that training a confidence network to rescale gradients, helps tackle the inherent noise attributed to the presence of a weak labeler.

We haven't considered λ as a function of time, and annealing λ over time may improve performance. Future work may also involve confidence networks with, momentum and adaptive learning rate based gradient update methods like Adam or RMSprop (Kingma and Ba, 2014). The glove embeddings capture the word contexts. Modelling curiosity (Venneti and Alam, 2018) in conjunction with such features may help capture user intent as well. Nevertheless, this requires a different set of experiments and benchmarks to thoroughly under-

[5]This includes the bias terms as well.

Model	Accuracy	Precision	Recall	ROC-AUC
Baseline (BiLSTMs)	0.982	**0.984**	0.980	**0.998**
Self-Attentive Network (SAN)	0.982	0.983	0.981	0.997
SAN + Gumbel Softmax (GS)	0.981	0.982	0.981	0.996
SAN + GS + Gaussian Filter (GF)	**0.983**	**0.984**	**0.982**	0.997

Table 4: Performance metrics against the existing baseline (A. Anand and Park, 2017), with the fully labeled *Headlines Dataset*. [Averaged over a 5-fold cross validation scheme]

Model	Accuracy			Precision			Recall		
	80%	50%	30%	80%	50%	30%	80%	50%	30%
Baseline (BiLSTMs)	0.980	0.966	**0.942**	0.979	0.967	**0.943**	0.981	0.966	**0.944**
SAN	0.979	0.966	0.944	0.978	0.965	0.945	0.980	0.967	0.942
SAN + RF	0.980	0.976	0.959	0.980	0.974	0.958	0.981	0.976	0.960
SAN + RF + GS	0.980	0.978	0.970	0.981	0.978	0.971	0.982	0.979	0.972
SAN + RF + GS + GF	0.981	0.977	0.975	0.982	0.978	0.976	0.981	0.977	0.977
SAN + RF + GS + GF + Confidence N/w	0.983	0.982	**0.980**	0.984	0.983	**0.980**	0.982	0.982	**0.981**

Table 5: Ablation study with variations in the proportions (80%, 50%, & 30%) of labeled data (*Headlines Dataset*). [Averaged over a 5-fold cross validation scheme]

Model	MSE			F1-Score			Accuracy		
	100%	50%	30%	100%	50%	30%	100%	50%	30%
Baseline	**0.033**	0.047	0.055	**0.683**	0.557	0.521	**0.856**	0.713	0.671
Proposed Model	0.034	**0.038**	**0.042**	0.679	**0.668**	**0.662**	**0.856**	**0.835**	**0.829**

Table 6: MSE, F1-Score and Accuracy metrics for the existing baseline (Zhou, 2017) & our solution on the test set, while using different proportions of set A (100%, 50% & 30%) as our labeled data.

stand the intricacies involved in such a mixture.

References

T. Chakraborty A. Anand and N. Park. 2017. We used neural networks to detect clickbaits: You wont believe what happened next! In *European Conference on Information Retrieval*, pages 541–547. Springer.

Roberto Alejo, Juan Monroy-de-Jesús, J. C. Ambriz-Polo, and J. H. Pacheco-Sanchez. 2017. An improved dynamic sampling back-propagation algorithm based on mean square error to face the multiclass imbalance problem. *Neural Computing and Applications*, 28(10):2843–2857.

Chidubem Arachie and Bert Huang. 2018. Adversarial labeling for learning without labels. *CoRR*, abs/1805.08877.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Abhijnan Chakraborty, Bhargavi Paranjape, Sourya Kakarla, and Niloy Ganguly. 2016. Stop clickbait: Detecting and preventing clickbaits in online news media. In *Advances in Social Networks Analysis and Mining (ASONAM), 2016 IEEE/ACM International Conference on*, pages 9–16. IEEE.

Mostafa Dehghani, Arash Mehrjou, Stephan Gouws, Jaap Kamps, and Bernhard Schölkopf. 2017a. Fidelity-weighted learning. *CoRR*, abs/1711.02799.

Mostafa Dehghani, Aliaksei Severyn, Sascha Rothe, and Jaap Kamps. 2017b. Learning to learn from weak supervision by full supervision. *CoRR*, abs/1711.11383.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pages 1050–1059.

A. Gianotto. 2014. Downworthy: A browser plugin to turn hyperbolic viral headlines into what they really mean. *downworthy.snipe.net*.

Ian J. Goodfellow, Yoshua Bengio, and Aaron C. Courville. 2016. *Deep Learning*. Adaptive computation and machine learning. MIT Press.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *CoRR*, abs/1611.01144.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Mu Li, Tong Zhang, Yuqiang Chen, and Alexander J. Smola. 2014. Efficient mini-batch training for stochastic optimization. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 661–670.

G. Loewenstein. 1994. The psychology of curiosity: A review and reinterpretation. *Psychological bulletin, 116(1):75*.

B. Stein M. Potthast, S. Kpsel and M. Hagen. 2016. Clickbait detection. In *European Conference on Information Retrieval*, pages 810–817. Springer.

M. Hagen M. Potthast, T. Gollub and B. Stein. 2017. The clickbait challenge 2017: Towards a regression model for clickbait strength. *Clickbait Challenge*.

K. Tsioutsiouliklis P. Biyani and J. Blackmer. 2016. 8 amazing secrets for getting more clicks: Detecting clickbaits in news streams using article informality. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 94–100.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Md Main Uddin Rony, Naeemul Hassan, and Mohammad Yousuf. 2017. Diving deep into clickbaits: Who use them to what extents in which topics with what effects? In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017, Sydney, Australia, July 31 - August 03, 2017*, pages 232–239.

Robert E. Schapire. 1990. The strength of weak learnability. *Machine Learning*, 5:197–227.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Trans. Signal Processing*, 45(11):2673–2681.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Ervin Teng, João Diogo Falcão, and Bob Iannucci. 2017. Clickbait: Click-based accelerated incremental training of convolutional neural networks. *CoRR*, abs/1709.05021.

Lasya Venneti and Aniket Alam. 2018. How curiosity can be modeled for a clickbait detector. *CoRR*, abs/1806.04212.

Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. 2016. Attention-based LSTM for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 606–615.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 2048–2057.

C. N. d. Santos M. Yu B. Xiang B. Zhou Z. Lin, M. Feng and Y. Bengio. 2017. A structured self-attentive sentence embedding. In *5th International Conference on Learning Representations*.

C. Dyer X. He A. J. Smola Z. Yang, D. Yang and E. H. Hovy. 2016. Hierarchical attention networks for document classification. In *Conference of the - Human Language Technologies*, pages 1480–1489. ACL.

Yiwei Zhou. 2017. Clickbait detection in tweets using self-attentive network. *In Proceedings of the Clickbait Challenge*.