Bringing Back Structure to Free Text Email Conversations with Recurrent Neural Networks

Tim Repke and Ralf Krestel

Hasso Plattner Institute, Potsdam, Germany
(tim.repke|ralf.krestel)@hpi.de

Abstract. Email communication plays an integral part of everybody's life nowadays. Especially for business emails, extracting and analysing these communication networks can reveal interesting patterns of processes and decision making within a company. Fraud detection is another application area where precise detection of communication networks is essential. In this paper we present an approach based on recurrent neural networks to untangle email threads originating from forward and reply behaviour. We further classify parts of emails into 2 or 5 zones to capture not only header and body information but also greetings and signatures. We show that our deep learning approach outperforms state-of-the-art systems based on traditional machine learning and hand-crafted rules. Besides using the well-known Enron email corpus for our experiments, we additionally created a new annotated email benchmark corpus from Apache mailing lists.

1 Introduction

Emails are an important part of day to day business communication, hence their analysis inspired research from a variety of disciplines. In Social Network Analysis, User Profiling, or Behaviour Analysis often only information contained in the well structured email protocol headers is used. However, a lot more information remains hidden in the free text body of an email, which contains additional meta-data about a discussion in the form of quoted messages that are forwarded or replied to.

In the early days of email communication, users followed clear rules, e.g. prefixing quoted text with angle brackets (>). Nowadays, due to the diversity of email programs, formatting standards, and the freedom to edit quoted text, identifying the different parts of a message body is a surprisingly challenging task. Email programs like Outlook, Thunderbird, or even online services such as Gmail, usually group emails into conversations and attempt to hide quoted parts. To this end, they try to match preceding emails by subject and sender, which fails in case the subject or quoted text was edited.

We propose a neural network based approach for extraction of the inherent structure in email text to overcome problems of error-prone rule-based approaches. This enables downstream tasks to work with much cleaner data and additional information by focusing on specific parts. Further we show improvements in flexibility and performance over earlier work on similar tasks.

Problem statement Our goal is to extract the inherent structure of free text emails containing a conversation thread composed of consecutive quoted or forwarded messages. Components of an email are referred to as *zones* similar to the definition used by Lampert et al [10]. We assume that a conversation thread is represented as a sequence of *client header* and *body blocks*. A pair of corresponding header and body is called *conversational part* or *message*.

In this context, client headers are blocks of meta-data automatically inserted by an email program, usually containing information on the sender, recipient, date, and subject of the quoted email. Generally the header indicates, whether the subsequent message body was forwarded or replied to by the text above. Bodies are the actual written messages, which on reply or forward are quoted below the newer message.

Message bodies can often be further separated into a *greeting* (such as a formal or informal address of the recipient at the beginning of the message), *authored text* (the actual message), *signoff* (closing words of the message), and a *signature* (containing contact information, advertising, or legal disclaimers). As emails with inline relies are usually copied, we consider a block of copied lines and responses as one body block.

We assume that each single line can be assigned to exactly one zone as Fig. 1 exemplary shows. In case of conflicts, the predominant or detailed type is used.

2 Related Work

Email corpora provide fascinating insights into human communication behaviour and therefore inspire research in many different areas. Datasets such as the Enron [9] or Avocado corpus [15] provide real world information about business communication and contain a mix of professional emails, personal emails, and spam. Ben Shneiderman published parts of his personal email archive for research [16]. Also popular is the 20 Newsgroups dataset [12] sampled from newsgroup postings in the early 90s, which we discard as it contains only few conversation threads. For the work at hand, we use the Enron corpus and emails we gathered from public email archives of the Apache Software Foundation[1].

A recent survey shows the diversity of email classification tasks alone [14]. Similarly interesting is the analysis of communication networks based on meta-data like sender, recipients, and time extracted from emails [1].

Models based on the written content of emails may get confused by automatically inserted text blocks or quoted messages. Thus, working with real world data requires normalisation of data prior to the problem at hand. Rauscher et al. [17] developed an approach to detect zones inside work-related emails where relevant business knowledge may be found.

In their work towards detecting emails containing requests for action, Lampert et al. [11] observed a relative error reduction by 40% when removing quoted sections of emails. Similar observations were made more recently predicting reply behaviour within the Avocado dataset [21].

[1] http://mail-archives.apache.org/mod_mbox/

From: Alice	*Sent:* Mon, 14 May 2001 07:15 AM
To: Bob, Brian	
Subject: RE: Telephone Call with Jerry Murdock	
Body	Thank you for your help.
Body	
Body/Signature	ISC Hotline
Header	03/15/2001 10:32 AM
Header	
Header	Sent by: Randi Howard
Header	To: Jeff Skilling/Corp/Enron@ENRON
Header	cc:
Header	Subject: Re: My "P" Number
Body	
Body/Greeting	Mr. Skilling:
Body	
Body	Your P number is P00500599. For your convenience, you can also go to
Body	http://isc.enron.com/ under Site Highlights and reset your password or
Body	find your "P" number.
Body/Signoff	Thanks,
Body/Signoff	
Body/Signoff	Randi Howard
Body/Signature	ISC HOTLINE
Body	
Header	From: Jeff Skilling 03/15/2001 10:01 AM
Header	
Header	To: ISC Hotline/Corp/Enron@Enron
Header	cc:
Header	
Header	Subject: My "P" Number
Body	
Body	Could you please forward my "P" number. I am unable to get into the XMS
Body	system and need this ASAP.
Body	
Body/Signoff	Thanks for your help.

Fig. 1: Example email with zones; consecutive blank lines reduced to one

Thread Reconstruction Another popular area of research is the reconstruction of graphs reflecting which message responds to another. Wang et al. propose baseline approaches based on temporal relationships [20]. There are also more advanced models that use sentence-level topic features to resolve a message graph using random walks [7]. Most recently, Tien et al. [19] proposed a novel convolutional neural network over a grid built by assigning roles to extracted entities. The latent graph is derived from the configuration with the highest coherence score. In our work however, we only focus on separating conversational parts within free text messages, not the actual reconstruction of the thread.

Email Zoning with Rules and Text Alignment. We identified three approaches to email zoning: rule based, text alignment, and machine learning.

The most naïve approach is to write specific rules that match commonly used patterns in email text. Talon[2] provides a sophisticated set of patterns to match most popular client header formats. The obvious downside is the lack of flexibility and that it's error-prone to changes.

Assuming a complete email corpus, a message in one user's outbox may be found in the inbox of other user(s). Likewise, quoted messages exist within

[2] https://github.com/mailgun/talon

the corpus as an original message from preceding communication. By finding overlapping text passages across the corpus, Jamison et al. managed to resolve email threads of the Enron corpus almost perfectly [6]. It has to be noted, that the claimed accuracy of almost 100% was only tested on 20 email threads.

In order to reassemble email threads, Yeh et al. considered a similar approach with a more elaborate evaluation reaching an accuracy of 98% separating email conversations into parts [22]. To do so, they rely on additional meta information in emails sent through Microsoft Outlook (thread index) and rules that match specific client headers. Thus, such an approach will not work on arbitrary emails, nor can it handle different localisation or edits by the user.

Contrary to approaches using text alignments, we don't assume a complete corpus. Our goal is to extract all information from only a single email archive or even a single email.

Machine Learning for Email Zoning. Another approach to email zoning uses machine learning with carefully designed features.

Carvalho and Cohn proposed Jangada [2], a system to remove quoted text and signature blocks from emails in the twenty newsgroup dataset [12]. They first classify emails to find those that contain quoted text or signatures and then classify each line individually using Conditional Random Fields (CRF) and sequence-aware perceptrons. Reported accuracies range from 97% to above 99%.

Other researchers applied Jangada to Hotmail emails and measured accuracies around 64% [4]. With some adaptation, they managed to extract five different zones (author text, signature, advertisement, quoted text and reply lines) with an average accuracy of up to 88%.

Lampert et al. developed the Zebra system [10] as a pre-processor to their previously mentioned work on requests for action [11]. Adversely to previous approaches, they use Support Vector Machines and therefore classify lines of an email into zones individually rather than considering a sequence of lines. For that, they describe graphic, orthographic, and lexical features to represent lines within their context reaching an average accuracy of 93% on the two-zone task and 87% on a nine-zone task. Comparing the performance by zone type, most problems are caused by signature lines (F-score around 60%), signoffs (70%) and attachments (69%). It was found, that adding contextual features didn't improve the performance [10]. Contrary to our objectives, Zebra only tries to identify the zones within the very last message within an email thread and rejects the rest as quoted text, whereas we aim to detect the zones across the entire email.

We compare results of our system described in Section 3 with Jangada and Zebra. We not only aim to improve upon those results, but also provide a system that is able to detect zones along the entire conversation thread contained in an email and not only the latest part. Furthermore, our system uses neural networks rendering expensive and potentially error-prone feature engineering obsolete. This way, even very small or incomplete datasets can be utilised for downstream tasks like social network analysis, speech act recognition and other research areas using email data.

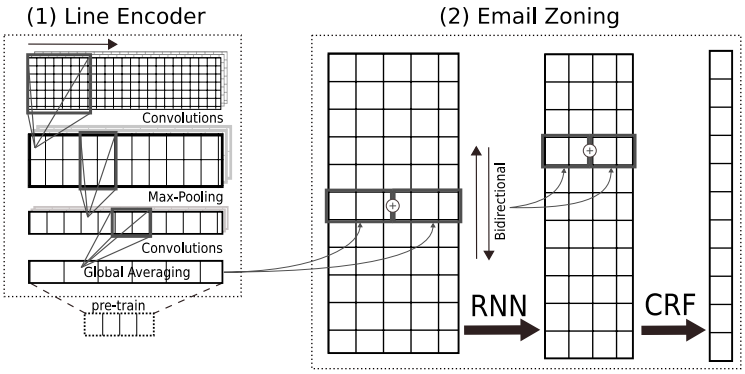

Fig. 2: Schematic model overview; Left side shows line embedding stage using the CNN approach, right side outlines email zoning model.

3 Segmentation of Emails

Systems for email segmentation that are discussed earlier are based on hand written rules to match common structures directly or use them as features for machine learning models. Such approaches will fail when client headers are localised, formats are changed, or quoted messages are edited by users or get corrupted.

In most cases it may seem obvious to the human eye how to segment an email into client headers and quoted text even though different or corrupted formats are used. However, even a sophisticated text parsing program will fail since client headers follow no standardised format. Usually lines start with attribute keywords such as "From:" or "Subject:", however their value may span multiple lines and use varying delimiters. This even makes it hard to detect the boundaries between header and body blocks, since one can not rely on the presence of keywords or well formed, deterministic schemas.

In this paper, we propose the *Quagga*[3] system based on neural network architectures. As shown in Fig. 2, emails are processed in two stages: the line encoding and the email zoning stage. In this section we describe he how email text is represented and how classifiers can be used as a reliable and robust preprocessor for a simple program to extract its inherent structure.

3.1 Representation of Email Data

In the initial stage of our system, the email text data is encoded into a low dimensional space to be used as input to the second stage as outlined on the left side in Fig. 2. The smallest fragments to be considered for email zoning are the lines in the email text. Lines are delimited by the newline character (\n), which may not necessarily be the same as wrapped lines displayed by an email

[3] The quagga is a subspecies of zebras. (https://en.wikipedia.org/wiki/Quagga)

program. Analysis of the annotated data shows that this granularity is sufficient for all header, body, and signature zones as was assumed by other research on similar tasks.

Each line is encoded as a sequence of one-hot vectors representing respective characters. We distinguish one hundred different case-sensitive alpha-numeric characters and basic ASCII symbols plus an out-of-scope placeholder. This is sufficient for all email corpora we looked at, where only a negligible portion of characters exceeds this set. We presume, that this could be adapted for applications with Cyrillic, Arabic or other alphabets.

Inspired by research on character-aware language models [8], we devised a recurrent and a convolutional neural network model. The recurrent model consists of a layer with varying number of gated recurrent units (GRU), where the last unit's output serves as a fixed size embedding of the line. The convolutional model uses two convolutional layers, which scan the sequence of characters in a line and are intertwined by max-pooling and global-averaging layers finally leading into a densely connected layer, where the number of neurons corresponds to the embedding size as shown on the left in Fig. 2.

In both models, the line representations are learnt in a supervised fashion. During training, a densely connected layer with softmax activation is appended so that a classifier can be trained to distinguish between lines of corresponding zone types. Optimal parameters of the topology such as the number of layers and embedding size are determined experimentally. A detailed analysis of embedding accuracy when limiting the length per line is found in Section 3.3.

3.2 Classification of Email Lines

A model in the first stage of our Quagga system learns line representations by classifying them into zones. That way however, the context in which a line appears is missing, resulting in less ideal performance on ambiguous or deceptive cases. Thus, we added a second stage to our system for sequence to sequence classification using a GRU-CRF model as outlined in the right part of Fig. 2, which takes a sequence of line encodings per email as input. Three of the five zone types only appear within message bodies, so we use two concatenated embeddings as input, where one is pre-trained using two- and the other with five-zone classification.

Best performance was achieved with a bidirectional GRU layer, which scans the lines from top to bottom and in reverse order and concatenates the hidden states of respective lines. In sequence to sequence classification, recurrent neural networks only consider the previous hidden state but neglect the actually predicted label sequence. We already observed small improvements by using a bidirectional layer over a unidirectional one, since each line's context reflects the previous and following lines. Like in language models [5, 13], the addition of a CRF to the output shows further performance gains.

Training both parts of the system as an entire model in one pass by directly connecting the encoder output layer to the second stage model's input lead to

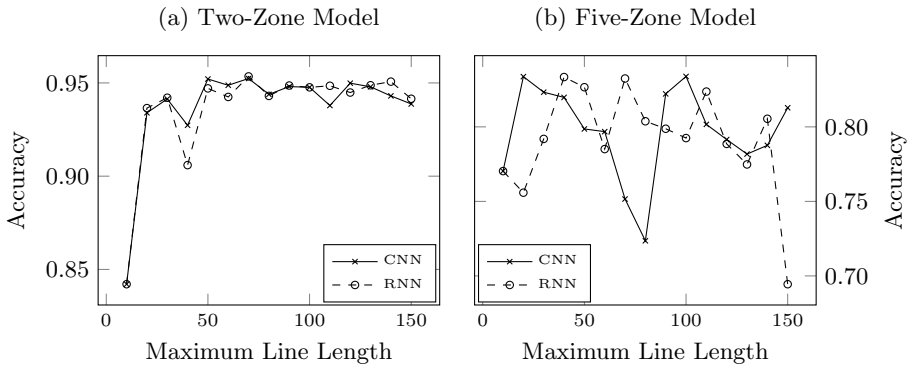

Fig. 3: Accuracy for increasing max line length using 32-dimensional embeddings

unstable results even after pre-training the line encoder. Therefore we train the model in the second stage of our system separately from the line encoding model.

The predicted sequence of zone types can be used to extract the conversational parts of an email and also separate the message from additional content such as signatures, greetings and signoffs. Consecutive lines with the same predicted zone type are aggregated into a block. Further processing inevitably requires making some assumptions about the general structure of emails. Based on the analysis of emails in the training data, we assume that a body proceeds a header block. Furthermore, we define that in-line replies with quoted parts belong to one message, which is not problematic, since this would usually only appear in Usenet-style emails, which use indicators like repeated > at the beginning of a line and therefore don't require sophisticated processing as presented here.

Small errors in the prediction can be fixed heuristically, for example a block with a single line classified as header containing only the "Subject:" keyword likely is either a false positive or belongs to another block nearby. The introduction of such rules could reduce the initial robustness and is omitted for the evaluation in this paper. In the scope of our work, we found that using the Quagga system as a pre-processor for finding related blocks significantly improves the accuracy of parsing rules for downstream tasks like constructing communication graphs from header blocks compared to a purely rule-based parser without pre-processing.

3.3 Selection of Model Parameters

In the description of the proposed model we highlighted adjustable parameters. This includes the model for line representations in the first stage, limiting the length of each line, and finding the ideal dimension of line embeddings. We base the model's topology configuration on the analysis of related models [8, 13, 5].

The ideal configuration for the line encoder model is determined through grid search across mentioned parameters. We record the accuracy of the convolutional

and recurrent approach for line encoding in Fig. 3. Note, that the convolutional model assumes a fixed size input, so shorter lines are zero-padded at the end. When evaluating the reported accuracy, one has to consider two things. First, this metric may not project down to the later stage of our system, and second, the line type distribution bias reduces the range of values to 0.81 (or 0.65) upwards.

We did not observe significant differences between embedding dimensions above 32, so we choose this dimensionality in favour of a less complex model. Most errors are caused by blank lines, which are usually classified as "Body". Results when training using two-zone classification are mostly stable for both models. The majority of lines in our training data are between forty and fifty characters long.

Both approaches for line representations seem to have their strengths and weaknesses. Since there is no clear winner, we continue only using the convolutional model in this work and fix the input length to 100 characters per line. We do so based on the argument, that one may want to process large corpora and prefer a faster system.

4 Experimental Setup

In this section we present an overview of the email datasets we used and discuss the sampling of emails to create an unbiased evaluation set. Further, we describe competing approaches that are used as baselines for comparison of our results. We also analyse model parameters and its robustness to changes in email text.

4.1 Dataset

We evaluate our proposed approach to email zoning on the Enron corpus [9] and emails gathered from public mail archives of the Apache Software Foundation[4] (ASF). Estival et al. [4] and Lampert et al. [10] discussed shortcomings in working with Usenet-style emails, leading us to refrain from using the twenty newsgroup dataset [12] as was done for the Jangada system. We found that more recent email threads from the ASF archives, especially those on mailinglists for users of different software projects, offer diverse formatting patterns.

Each dataset is divided into three subsets for training, validation, and testing. Emails are sampled at random from their respective original dataset and put into one of those subsets. To ensure representative results that are not biased by author or domain, sampling per subset is restricted to distinct mailboxes (Enron) or mailing lists (ASF). The ASF dataset was compiled by randomly selecting emails from the *flink-user*, *spark-user*, and *lucene-solr-user* mailing list archives.[5]

Table 1 shows an overview of the selected dataset and the expected number of messages to be extracted. Prior heuristic analysis of the Enron corpus estimated

[4] http://mail-archives.apache.org/mod_mbox/

[5] Annotated datasets and code can be found at https://github.com/TimRepke/Quagga

Table 1: Annotated Datasets in Numbers

	Enron			ASF		
	Train	Test	Eval	Train	Test	Eval
Emails	500	200	100	350	100	50
Individual messages	1048	474	233	934	226	108
Average length of threads	3.5	3.6	3.5	3.5	3.7	3.1
Number of signatures	103	58	26	76	13	5

60% of emails to contain conversation threads [9], which is close to our annotated data. On average an email has two parts with 20 lines per message. Only a few messages contain a signature, which on average are six lines long.

4.2 Competing Approaches

We compare our proposed model for extracting zones from emails against several other approaches. Most notably, Jangada [2] and Zebra [10] are reimplemented with slight modifications to fit the more refined problem statement. Both systems originally are intended to distinguish lines within an email, which are not part of the latest message of that thread. Clearly that deviates from our goal to extract *all* individual parts and detect zones with additional detail within those. Since the systems are supposed to detect zones within the first part of the email, their features and underlying models should in principle also work on our task.

The source code for Jangada is freely available on the author's web page[6]. We used the source code as a basis for an implementation in Python and the originally used model for sequence labelling, which is part of the MinorThird Library[7]. For the extraction of signatures, Jangada originally only considers the last ten lines of an email. In our implementation, the perceptron performs a multi-class classification along all lines of the email corresponding to zone types defined earlier. The model is trained with window-size 5 for 40 epochs.

The Zebra project web page[8] does only provide annotated data, but not the system's source code. Gossen et al. implemented[9] it for their work on classification of action items in emails [18]. We used that as a guideline for our adapted Python implementation. The SVM is trained for a maximum of 200 iterations in a one-versus-rest fashion for multi-class classification using RBF kernels.

We use a selection of features from both models as input for a recurrent neural network with two GRU layers [3], which we will refer to as *FeatureRNN*. The above models are baselines for the comparison to our proposed Quagga system as described in Section 3 using a convolutional model as line encoder with fixed input sizes of 100 characters per line.

[6] http://www.cs.cmu.edu/ vitor/software/jangada/
[7] http://minorthird.sourceforge.net/
[8] http://zebra.thoughtlets.org/zoning.php
[9] https://github.com/gerhardgossen/soZebra

Table 2: Classifying Emails into Zones (Precision/Recall/Accuracy)

Approach	Zones	Enron	ASF
Jangada[2]	2	0.89 / 0.88 / 0.88	0.97 / 0.97 / 0.97
Zebra[10]	2	0.66 / 0.25 / 0.25	0.88 / 0.18 / 0.18
FeatureRNN	2	0.98 / 0.98 / 0.97	0.97 / 0.95 / 0.94
Quagga	2	**0.98 / 0.98 / 0.98**	**0.98 / 0.98 / 0.98**
Jangada[2]	5	0.82 / 0.85 / 0.85	0.90 / 0.92 / 0.91
Zebra[10]	5	0.60 / 0.25 / 0.24	0.81 / 0.20 / 0.20
FeatureRNN	5	0.92 / 0.75 / 0.75	0.90 / 0.60 / 0.60
Quagga	5	**0.93 / 0.93 / 0.93**	**0.95 / 0.95 / 0.95**

5 Results

In this section we compare Quagga to similar systems found in related work. To get a good understanding of the versatility, we not only look at the results shown in Table 2, but also consider the robustness against noise or otherwise changing data as well as how many training samples are required to get good results.

We were not able to reproduce reported accuracies of Zebra [10], which given the nature of the features and individual classification of lines disregarding their position and context in an email was expected. Jangada uses more general features and looks at a sliding window of lines and we got close to reported accuracies, especially for the ASF dataset, which is closer to the twenty newsgroup data the authors used [10]. Overall, our system shows very good performances and seamlessly adapts to other datasets without problems.

Number of Training Samples Complex neural network based machine learning models require lots of training samples to reliably proficiently solve a given task. We limited the number of Enron emails shown to the network during training down to 10% of the Enron training set, corresponding to 50 emails and then measured the performance whilst continuing to add training samples up to all 500 emails. The model trained with the least data in this scenario only lags behind around 1% in accuracy compared to a model trained on all data in both the two- and five-zone task.

Cross Corpus Compatibility Ideally, a system like Quagga would be trained once and work well on arbitrary emails. The Enron and ASF datasets show more differences than there are in samples in the training and testing data of one corpus. Thus we trained Quagga on Enron and tested it on ASF emails and vice versa. We observe, that by training on ASF emails, Quagga does not generalise as well to emails from he other corpus (Accuracy: 0.86 or 0.80, for two- or five-zones) as the other way around (Accuracy: 0.94 or 0.86). Compared to the results when training and testing using emails from the same base dataset, this results in a decrease of performance of around 4-10%.

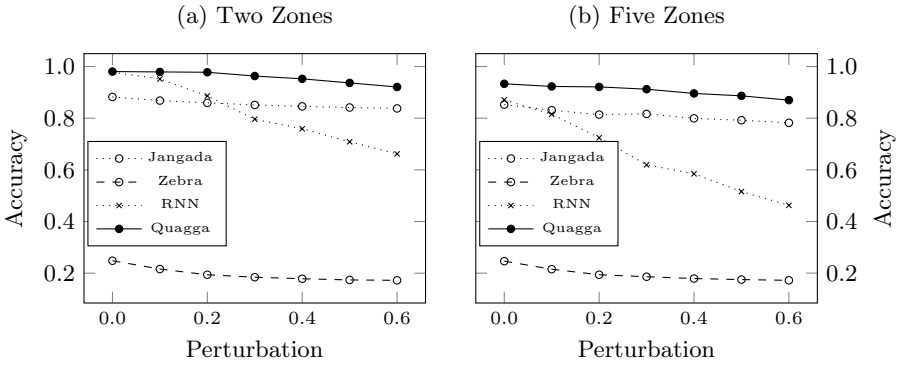

Fig. 4: Robustness against perturbation for the Enron test set

Robustness to Noise Our hypothesis is, that a model which learns meaningful features itself is more robust towards changes to the email text as hard coded rules responding to specific keywords or patterns. To show the flexibility of our model, we first introduce the notion of a perturbation threshold $\rho \in [0, 1)$. Before passing an email to a model, a function iterates over each character and with probability ρ edits, removes, or duplicates it. Training was performed on uncorrupted data only.

The robustness of each model against increasing perturbation is shown in Fig. 4. Relatively, the drop in performance is the same the for two- and five-zone task, although at different absolute accuracies. Quagga doesn't seem to be affected up to $\rho = 0.2$ and keeps producing reliable results even at higher perturbation thresholds. Surprisingly, also Jangada is not influenced significantly by the introduction of perturbation. As opposed to Zebra and FeatureRNN, it is using more features related to small patterns or proportions of types of symbols, whereas the others depend on more complex patterns which are more error prone to change.

6 Conclusion and Future Work

In this work we presented a reliable and flexible system for finding the inherent structure of an email containing multiple conversational parts. In the first stage, the system uses a convolutional neural network to encode lines of an email which are used by a GRU-CRF to predict a sequence of zone types per line reaching accuracies of 98%. Compared to similar models, we show significant improvement and especially seamless adaptation to other datasets as well as robustness against corrupted data.

Research based on email data can largely benefit from this system by pre-processing the text and focus downstream algorithms on relevant parts of an email like client headers or the actual text cleaned of irrelevant parts.

In addition to the Quagga system, we provide a detailed annotation of a subset of the Enron corpus that can directly be used for building communication networks without further parsing including linked person aliases and, if present, contact details from email signatures as well as the new ASF corpus.

References

1. Bonchi, F., Castillo, C., Gionis, A., Jaimes, A.: Social network analysis and mining for business applications. TIST 2(3) (2011)
2. Carvalho, V., Cohen, W.: Learning to Extract Signature and Reply Lines from Email. In: CEAS (2004)
3. Cho, K., van Merrienboer, B., Bahdanau, D., Bengio, Y.: On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. CoRR (2014)
4. Estival, D., Gaustad, T., Pham, S., Radford, W., Hutchinson, B.: Author profiling for English emails. In: Conference of the Pacific ACL (2007)
5. Huang, Z., Xu, W., Yu, K.: Bidirectional LSTM-CRF Models for Sequence Tagging. CoRR (2015)
6. Jamison, E., Gurevych, I.: Headerless, Quoteless, but not Hopeless? Using Pairwise Email Classification to Disentangle Email Threads. In: RANLP (2013)
7. Joty, S., Carenini, G., Ng, R.T.: Topic segmentation and labeling in asynchronous conversations. Artificial Intelligence Research 47, 521–573 (2013)
8. Kim, Y., Jernite, Y., Sontag, D., Rush, A.: Character-Aware Neural Language Models. CoRR (2015)
9. Klimt, B., Yang, Y.: The Enron corpus: A new dataset for email classification research. Machine learning: ECML (2004)
10. Lampert, A., Dale, R., Paris, C.: Segmenting Email Message Text into Zones. In: EMNLP (2009)
11. Lampert, A., Dale, R., Paris, C.: Detecting Emails Containing Requests for Action. In: Human Language Technologies. ACL (2010)
12. Lang, K.: Newsweeder: Learning to filter netnews. In: Twelfth International Conference on Machine Learning (1995)
13. Ma, X., Hovy, E.H.: End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. CoRR (2016)
14. Mujtaba, G., Shuib, L., Raj, R., Majeed, N., Al-Garadi, M.: Email Classification Research Trends: Review and Open Issues. IEEE Access (2017)
15. Oard, D., Webber, W., Kirsch, D., Golitsynskiy, S.: Avocado research email collection. Linguistic Data Consortium (2015)
16. Perer, A., Shneiderman, B.: Beyond threads: Identifying discussions in email archives. Tech. rep., MUC (2005)
17. Rauscher, F., Matta, N., Atifi, H.: Context Aware Knowledge Zoning: Traceability and Business Emails. In: IFIP Workshop (2015)
18. Scerri, S., Gossen, G., Davis, B., Handschuh, S.: Classifying Action Items for Semantic Email. In: LREC (2010)
19. Tien Nguyen, D., Joty, S., El Amel Boussaha, B., de Rijke, M.: Thread reconstruction in conversational data using neural coherence models. In: Neu–IR (2017)
20. Wang, Y.C., Joshi, M., Cohen, W.W., Rosé, C.P.: Recovering implicit thread structure in newsgroup style conversations. In: ICWSM (2008)
21. Yang, L., Dumais, S., Bennett, P., Awadallah, A.: Characterizing and Predicting Enterprise Email Reply Behavior. In: SIGIR (2017)
22. Yeh, J., Hamly, A.: Thread Reassembly Using Similary Matching. In: CEAS (2006)