An On-device Deep-Learning Approach for Attribute Extraction from Heterogeneous Unstructured Text

Mahesh Gorijala **Aniruddha Bala** **Pinaki Bhaskar** **Krishnaditya Vikram Mupparthi**
Advanced Technology Lab (ATL)
Samsung R&D Institute India - Bangalore (SRI-B)
{m.gorijala, aniruddha.b, pinaki.b, krish.aditya, vikram.m}
@samsung.com

Abstract

Mobile devices, with their rapidly growing usage, have turned into rich sources of user information, holding critical insights for betterment of user experience and personalization. Creating, receiving and storing important information in the form of unstructured text has become a part and parcel of daily routine of users. From purchase deliveries in Short Message Service (SMS) or Notifications, to event booking details in Calendar applications, mobile devices serve as a portal for understanding user interests, behaviours and activities through information extraction. In this paper, we address the challenge of on-device extraction of user information from unstructured data in natural language from heterogeneous sources like messages, notification, calendar etc. The issue of privacy concern is effectively eliminated by the on-device nature of the proposed solution. Our proposed solution consists of 3 components – A Naïve-Bayes based classifier for domain identification, a Dual Character and Word based Bidirectional Long Short Term Memory (Bi-LSTM) and Conditional Random Field (CRF) model for attribute extraction and a rule-based Entity Linker. Our solution achieved a 93.29% F1 score on five domains (shopping, travel, event, service and personal). Since on-device deployment has memory and latency constraints, we ensure minimal model size and optimal inference latency. To demonstrate the efficacy of our approach, we have experimented on CoNLL-2003 dataset and achieved comparable performance to existing benchmark results.

1 Introduction

With an estimated 3.5 billion active users or about 80% of all mobile subscribers, Short Message Service (SMS) was the most widely used communication application in the past few years [1]. Even with the advent of social media and messenger applications, communication and information storage in digitised form are vastly prevalent via SMS, notifications, calendar invites and mail. Some examples we readily see are casual conversations with a friend over SMS, online shopping related notifications and event booking details, just to name a few.

According to 2020 Annual report by CTIA [2] , there were 2.1 trillion text messages exchanged worldwide, an increase of 52 billion messages since 2019. According to SMS marketing statistics for 2020/2021 reported by FinancesOnline [3] , 98% of SMS are opened compared to only 20% of emails and 95% of the read SMS are responded to within 3 minutes of delivery. Moreover, SMS is still the most powerful marketing tool for businesses with 75% of customers preferring receiving offers via SMS. The CTR for text messages is much higher (9.18%), compared to other marketing channels such as Google Adwords (1.91%) and Facebook (0.90%).

Apart from SMS and notifications, researchers have investigated other potential data sources like calendar, email, user utterances and communication logs for extracting information. In fact, many establishments are making use of the personal knowledge extracted from different sources on smartphones to provide better service. For example, Google extracts and summarizes travel, event and accommodation reservation information from emails [4] . However, most of the published literature is focussed on singular sources of information and/or certain domains of interest like bio-medical,

[1] https://en.wikipedia.org/wiki/SMS

[2] www.ctia.org/news/report-2020-annual-survey-highlight

[3] https://financesonline.com/sms-marketing-statistics/

[4] https://developers.google.com/gmail/markup/reference/#reservations

Proceedings of the 18th International Conference on Natural Language Processing, pages 573–582
Silchar, India. December 16 - 19, 2021.

Data Source	Information Extracted	Inference Drawn
Messages	Shopping, travel and financial activities, event attendance, service availed etc.	Preferred vendors, products, venues, payment modes etc.
Calendar	past and upcoming events and occasions	Preferred relations
Call & Message logs	caller, callee, message sender, receiver details	Frequent caller, callee, message sender and receiver
Notifications	All the above details and activities	User's details, preference and interest

Table 1: Particular data items extracted and high-level inferences drawn from the data sources.

events, etc.

In this paper, we address the challenge of on-device extraction of user information from unstructured data in natural language from heterogeneous sources, which include SMS, notifications, calendar etc. Our proposed system offers a unique method for efficient on-device functionalities. We achieve 93.29% F1 score on five domains (shopping, travel, event, service and personal). The system is implemented as a service on the device. The information that is obtained from these data sources with a preliminary analysis and the high-level inferences sought from it, are summarized in Table 1. The abundance of personal information on smartphones can hence be safely utilized for many apps like recommender systems, virtual personal assistants, on-device content presentation, to provide better services to end users. This provides a holistic view about the user encompassing users' behaviours, interests, activities, etc.

A key consideration is the constant ongoing conflict between the service provider's desire to track the consumer and the consumer's concern for the privacy of their data. This issue is effectively addressed by the on-device nature of the proposed system, letting the user enjoy its benefits conveniently as the data processing is limited to local environment.

Some of the features afforded by this new dimension of user's data, which enables personalized device intelligence, are as follows:

- User's attention can be proactively drawn to offers and discounts regarding the products of only the categories they wish to purchase, filtering all the annoying spam.
- Enable simplified interaction with smart assistant
- Event reminders can be triggered appropriately
- Convenient grouping or reordering of SMS/ Notification/ Calendar data according to user preferences
- Assist in better planning of activities, for instance, booking airport cab with prior knowledge of user's travel plans
- Recommender services based on understanding of user's shopping behaviour or preferred types of events (like concerts, sports matches, photography, art etc.)

2 Related Work

Digital communication devices continue to offer a growing variety of personalized services to enhance user experience. This is facilitated by increased access and extraction of user information available in both structured and unstructured forms. Structured data, generally consisting of text entered in template fashion or in any pre-defined format (like date, zip code etc.), can be conveniently processed whereas unstructured text (like Short Message Service (SMS), Notifications, Calendar events etc.) poses multiple interesting challenges.

Firstly, apart from emanating from heterogeneous sources on the device, unstructured data on mobile devices does not always conform to grammatical correctness, rendering it difficult for most of the existing Natural Language Processing (NLP) techniques better suited for formal grammar. Secondly, most of the advanced information extraction techniques demand server-based deployment, raising privacy concerns of user data storage on cloud. There is limited exploration on on-device information extraction from unstructured text, befitting

its memory and latency constraint requirements. Thirdly, owing to the special nature of data in consideration, there is a lack of standardised benchmark datasets. Most of the previous works have shown a significant amount of research effort being directed to collection, curation and pre-processing of short-text corpus. After data procurement comes the daunting task of annotation based on the identified guidelines of entities and attributes relevant to each domain of interest.

There is some pre-existing work on domain classification and information extraction from email, SMS, notifications and social media text. Most of the previous works handling SMS are primarily focused on spam-filtering or certain rudimentary levels of classification. Almeida et al. (2011) and Cormack et al. (2007), limit the task to binary classification of SMS as spam or non-spam. Dewi et al. (2017), explore the possibility of multi-class classification of messages into 4 categories with limited data instances. They achieve best results with logistic regression. In comparison, we categorize messages into 6 classes and perform further information extraction.

With regard to information extraction from short texts like SMS and notifications, traditionally various approaches have been investigated including use of POS taggers, regular expressions, hidden Markov models (HMM), logistic regression, specific syntactic parsers or a combination of the above. Jiang et al. (2010) investigate the extraction of named entities related to events or activities from Chinese SMSes in handsets, using Hidden Markov models (HMM). Although their method achieves a lower F-score on a small SMS corpus of 1,000 messages, the authors significantly reduce the memory consumption. Polifroni et al. (2010) implement logistic regression to recognize name, date, location and time entities from messages. Their reported F-scores for names and locations reaches 88 on an individual word basis, but they do not report on computational or memory resources required of their approach or exact corpus size. Cooper et al. (2005), exploit the syntactic structure in messages and used pattern matching for extraction. Since pattern matching is not robust to variations in data, Ek et al. (2011) complement pattern matching with a logistic regression based classifier.

Recent works on SMS and notifications involves the use of deep learning models for information extraction. Vatsal et al. (2020) implement a hybrid hierarchical LSTM-CNN architecture for SMS classification and then use class specific entity parsers based on pattern matching. Li et al. (2018) use the insight that notifications are formatted using templates. Templates are extracted using longest common subsequence mining and then clustered using DBSCAN algorithm. Template semantic rules are then generated using a Bi-LSTM network.

We believe ours is the first work that provides a generalized deep learning architecture for information extraction from multiple unstructured data sources. We also categorize inputs into multiple domains and link the attributes to pre-existing entities in the database. Our system pipeline has been designed to cater to multiple applications such as customization services, recommender systems, knowledge base population, etc. requiring the holistic understanding of users.

3 Proposed Methodology

The proposed information extraction pipeline consists of 3 major components – Domain Classifier, Attribute Extractor and Entity Linker. A pictorial representation of the pipeline is depicted in Fig. 1.

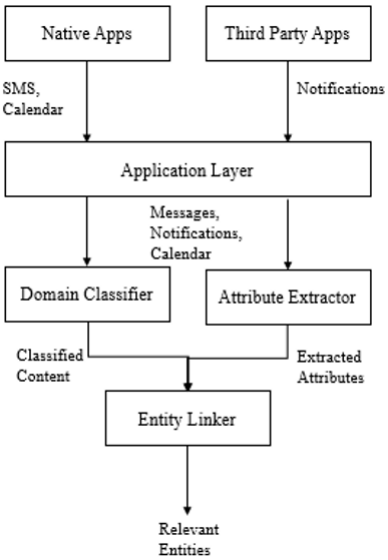

Figure 1: System Overview

3.1 Data Preprocessing

We converted messages and notifications into a more generalised format by using pattern matching. All date and time variations were mapped to ⟨DATE⟩ and ⟨TIME⟩ tokens respectively and currency values were mapped to ⟨CURRENCY⟩ tokens. All other numeric values were replaced by ⟨NUM⟩ and alphanumeric values were converted to

⟨ALPHANUM⟩ token. Website URLs were identified and replaced by ⟨URL⟩ token. For e.g. the input sentence

> dispatched : your package with philips dj shl3000 / 00 over - ear headphone (blue) will be delivered on or before wed , june 29 . track at www.amzn.in/track

will be processed as

> dispatched : your package with philips dj ⟨ALPHANUM⟩ / ⟨NUM⟩ over - ear headphone (blue) will be delivered on or before ⟨DATE⟩ , ⟨DATE⟩ . track at ⟨URL⟩

3.2 Domain Classifier

This module classifies the unstructured part of the data into one of a set of predefined domains. For this work, the predefined set includes Shopping, Travel, Event, Service, Personal & Spam. This classification allows the identification of user's domain-wise preferences, which increases the accuracy of the recommendations / ranking generated by applications using extracted information. Since the domains are very distinct, we propose a hybrid Naïve Bayes model for this simple text classification task. The block diagram for proposed approach is shown in Fig. 2. We augmented the standard Naïve Bayes model with an n-gram language model to effectively capture the inherent template-like structures in the data. The probabilities were computed based on tf-idf of tokens along with its length and the sender (in the case of messages and notifications).

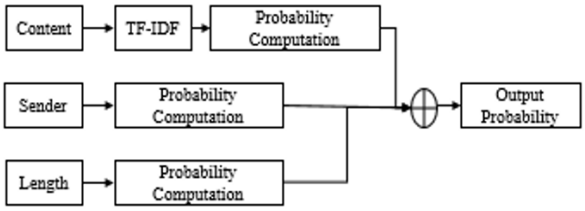

Figure 2: Domain Classifier Architecture

The domain classifier is used in the downstream task of triplet generation. For e.g. for an SMS where a token is tagged as Start Date or End Date, we will use the domain classifier output to decide the property name before adding the triple to the knowledge graph. For instance, if the domain of the SMS is shopping then we would know that the tagged date corresponds to product delivery date. We also tried to use the domain information as an input to our attribute extraction model, however, it did not improve the overall performance.

3.3 Attribute Extractor

This module extracts a predefined set of attributes (listed in Table 3) from the unstructured part of the data, which contain the pieces of information about the event / activity being conveyed by the data. Attribute extraction is modelled as a sequence labelling task. We implement a dual character and word embedding based Bi-LSTM (Hochreiter and Schmidhuber, 1997) followed by a CRF (Lafferty et al., 2001) trained on the sequence labelled set of messages, notifications and calendar. In the training dataset, all attribute tokens in a training sample are appropriately marked with one of the a) "B" for beginning, b) "M" for middle and c) "E" for end token, followed by the attribute type tags. Non-attribute tokens are marked with "O" (other) tag. E.g. "Your order for Samsung Galaxy S20 will be delivered today" is marked as "O O O B-Product M-Product E-Product B-Status M-Status E-Status B-EndDate". All tokens in the training dataset with frequency greater than 5 are included in vocabulary and the remaining tokens are substituted by the ⟨UNK⟩ token. We also create a character vocabulary required for generating character based word embeddings. Using the two vocabularies, we generate word and character lookup tables that are required for tokenization of input.

Fig. 3 describes the process of generating embeddings for each token in the input sentence. The input sequence is first encoded using the word lookup table and then passed into an embedding layer, which gives us W_{emb}. Each character is then considered as a token and encoded using the character lookup table. The output is passed into bidirectional $LSTM_{char}$ which generates forward and backward representations. These are then concatenated to give character based word embedding. Ultimately, the word embedding and the character based word embedding are concatenated to give the final token embedding. The computations performed inside LSTM cells are as follows:

$$i_t = \sigma(w_i[h_{t-1}, x_t] + b_i) \quad (1)$$

$$f_t = \sigma(w_f[h_{t-1}, x_t] + b_f) \quad (2)$$

$$o_t = \sigma(w_o[h_{t-1}, x_t] + b_o) \quad (3)$$

$$g_t = tanh(w_g[h_{t-1}, x_t] + b_g) \quad (4)$$

$$c_t = f_t * c_{t-1} + i_t * g_t \quad (5)$$

$$h_t = o_t * tanh(c_t) \quad (6)$$

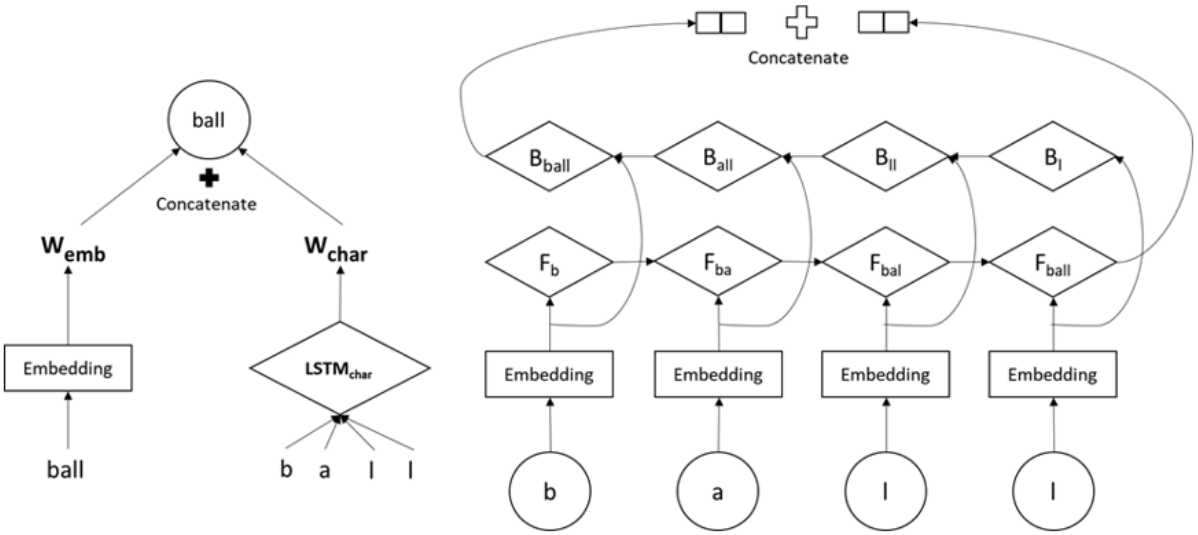

Figure 3: Word Embedding Generation in Dual Bi-LSTM: The representation of a word is formed by concatenating the word embedding and the character level representation of the word from char LSTM.

Where i_t, f_t, o_t and g_t are input, forget, output and cell gates respectively, x_t is input at time step t, h_t is hidden state and c_t is cell state. The hidden states at final time step are considered as representation of input sequence. The proposed attribute extractor model is depicted in Fig. 4. The input token embeddings are fed into the Bi-LSTM encoder followed by a fully connected layer. This generates emission scores, which represent likelihood of word being a certain tag. The role of the CRF layer is to model the joint likelihood of the entire tag sequence. This is achieved by calculating the transition scores, which represent the likelihood of word being a certain tag given the previous word was a certain tag. For decoding, Viterbi algorithm is used to find the tag sequence with maximum likelihood.

3.4 Entity Linker

This module links the entities identified by the attribute extraction module to an appropriate entity in the existing database having either the same or different name. It also processes the "Sender" information (for messages and notifications) and "Date" information (for calendar events) and accordingly adds entities if they weren't identified by the attribute extractor from the main content. As entity linking module gets diverse attributes as its input, it's implemented with different approaches.

- We use off-the-shelf string matching algorithms such as FuzzyWuzzy [5] based on Levenshtein Distance and phonetic algorithms such as Soundex [6] for linking the vendor attribute.
- We use an ontology and a predefined set of rules to match Source Location, Destination, Travel Mode, Travel Class, Event Type, Service Type, Status, Relationship and Occasion attributes.
- In case of Start Date, Start Time, End Date and End Time, we identify all possible variations in our data and map them to a standard format using pattern matching.
- Some attributes like ID, Product, Vehicle Number, Event Name and Amount are left unmatched.

4 Dataset

For data collection we sent out an organization wide broadcast seeking voluntary participation from users with diverse demographics . For this purpose, the users were required to install an application developed by the team. The application masked the user's private information such as name, contact number, financial details etc. and allowed the user the option to filter messages before sharing.

We collected $\sim 90K$ $(90,811)$ messages, $\sim 55K$ (54990) notification and $\sim 1K$ calendar

[5] https://pypi.org/project/fuzzywuzzy/
[6] https://pypi.org/project/soundex/

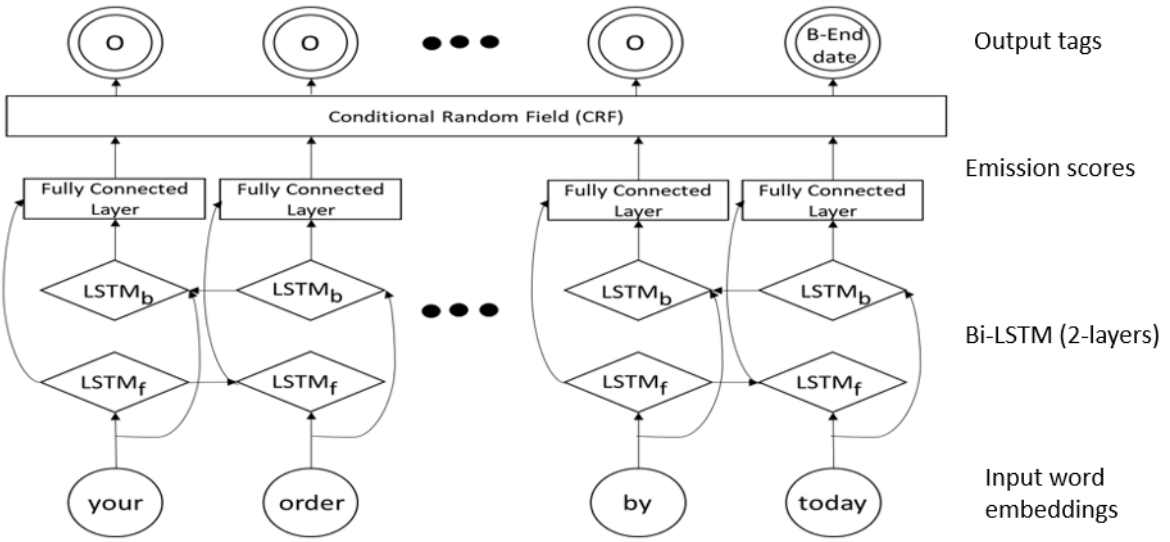

Figure 4: Attribute Extractor Architecture

Dataset	Domains	Message	Notification	Calendar
Training Set	Shopping	5715	621	0
	Travel	2982	353	77
	Event	2609	467	217
	Service	2508	571	0
	Personal	3865	5309	235
Test Set	All five	2000	500	500

Table 2: Data Distribution Across All Domains

data. We then filtered the collected data to get $\sim 17k$ $(16,959)$ relevant messages, $\sim 7K$ (7321) relevant notifications and 720 calendar data and classified them into 5 domains (Shopping, Travel, Event, Service, Personal). The data distribution over these domains is shown in Table 2.

The collected data was then clustered based on similar templates for the ease of annotation. For e.g. the product delivery messages from Amazon follows a certain template with only change being in certain fields such as product name, delivery agent contact etc. We then chose an exemplar from each of these clusters and asked annotators to annotate each of the words in the text with the appropriate tag. We then curated a test set from $\sim 9k$ relevant data collected from different individuals. This was kept separate from the training data and was handpicked to include unique instances.

Table 1 in appendix displays one sample instance per domain and the identified relevant attributes while Table 2 covers the entity linker output for the previously chosen sample instances along with relevant fields like Sender and Date. The list of all relevant attributes was generated by a comprehensive analysis of collected data and usefulness of information contained in the data source. The distribution and relevant domains for each attribute is given in Table 3.

Attribute	Relevant Domains	Count
ID	Shopping, Travel, Event, Service	3652
Status	Shopping, Travel, Event, Service	8751
Vendor	Shopping, Travel, Event, Service	3961
Product	Shopping	2910
Start Date	Travel, Event	2749
End Date	Shopping, Travel, Event, Service	2164
Start Time	Travel, Event, Service	2913
End Time	Travel, Event, Service	1458
Travel Mode	Travel	1835
Travel Class	Travel	840
PNR	Travel	1553
Vehicle Number	Travel	1829
Source Location	Travel	2117
Destination	Travel, Event	2402
Event Name	Event	639
Event Type	Event	131
Service Type	Service	143
Amount	Shopping, Travel	1203
Relationship	Personal	41
Occasion	Personal	76

Table 3: Distribution of Attributes and Relevant Domains

5 Experiments and Results

5.1 Evaluation Metrics

We performed evaluation on the test set (described in Table 2). We used weighted F1 score, precision and recall to evaluate performance of our proposed pipeline. Since our system focuses on on-device extraction, latency and memory usage are also critical

metrics. The total model size (including embedding size) is also reported for every model. The latency measurements were done by averaging over randomly picked 100 data points.

5.2 Domain Classifier

We compare 3 different models for domain classification and the model parameters are given in Table 4.

- **Hybrid Naïve Bayes Classifier:** This is the proposed domain classifier. We computed Tf-Idf of unigram, bigram, trigram and quadgram tokens and used these to compute class probabilities.
- **Deep Neural Network (DNN):** We generate token embeddings using a filtered version of Glove embeddings to reduce memory usage. The model accepts these token embeddings as input and generates a distribution over the predefined set of domains.
- **Convolutional Neural Network (CNN):** Like in the previous model, we generate token embeddings using filtered version of Glove embeddings. These token embeddings act as input for convolutional layers that extract feature maps. This is further passed into Max-pooling layer and then a final linear layer which gives output distribution.

From Table 5, we can see that all the models gave comparably high F1 scores on the test data with the Naïve Bayes classifier just edging the other two. Contrary to expectations, the deep learning based approaches do not outperform the simpler Naïve Bayes model. This is because the domains do not overlap and have very distinct samples and hence, do not need a complex model for accurate distinction. Since all computed latencies are very low, we give preference to memory usage during selection.

5.3 Attribute Extractor

We compare the performance of 3 deep learning models with different encoders followed by a CRF decoder.

- **Bi-LSTM + CRF:** This is the standard approach for sequence labelling. The Bi-LSTM encodes the input and the CRF acts as the decoder. This model uses only the word level features of the sentence as input.

DNN	Number of layers	2
	Number of Hidden Units	50, 20
	Dropout value	0.5
CNN	Number of 2D convolutional layers	2
	Number of filters	64, 32
	Kernel dimensions	5, 3
	Dropout value	0.5
Optimizer & learning rate		Adam, 0.001
Train - validation split		80 - 20

Table 4: Domain Classifier Model Parameters

Metric	Näive Bayes	CNN	DNN
F1 Score	**0.9596**	0.9551	0.9561
Precision	**0.9713**	0.9472	0.9487
Recall	0.9482	0.9632	**0.9637**
Model Size (KB)	2680	378	983
Embedding Size (KB)	NA	4636	4636
Total Memory (KB)	**2680**	5014	5619
Latency (ms)	91.7	**14.7**	19.8

Table 5: Domain Classifier Results

- **Dual Bi-LSTM + CRF:** This is the proposed model. It captures the character level information along with word level features.
- **Transformer + CRF:** We use a multi-headed transformer encoder followed by a CRF decoder.

The model details for each aforementioned approach are given in Table 6. We also experiment addition of the domain classifier output as input into the attribute extraction model. The domain is added as an extra input to the message/notification and then the combined input is fed into the embedding layer.

We see from Table 7 that our proposed Dual Bi-LSTM encoder just outperforms the standard Bi-LSTM. This verifies the ability of character embeddings to capture greater morphological diversity, which is especially visible in SMS and notifications. We also experiment with the transformer model for the attribute extraction task. Owing to the huge model size of pre-trained transformers like BERT, we limit ourselves to a custom two layer transformer model which is trained from scratch. The inferior performance of the transformer model compared to the Bi-LSTM model can be attributed to over-parameterization and lack of pre-training. We also observe that adding domain input slightly worsens the performance. This is because our attributes are structured such that similar attributes across different domains are considered as one. E.g. Delivery Date, Event Date and Service Date are all

considered as End Date. Hence, adding the domain input adds to the complexity of input and we observe slight drop in performance.

We also run our models on the popular English NER dataset - CoNLL2003. It contains four different named entities: PERSON, LOCATION, ORGANIZATION, and MISC. The dataset consists of 14000 training samples, 3200 validation samples and 3500 test samples. The data is tokenized using the same preprocessing as mentioned in 3.1.

BiLSTM + CRF	Number of layers	2
	Number of Hidden Units	128
	Embedding Dimension	128
Dual Bi-LSTM + CRF	Number of layers	2
	Number of hidden units	128
	Word embedding dimension	128
	Character embedding dimension	64
	Number of char-LSTM hidden units	64
Transformer + CRF	Number of layers	2
	Positional encoding dimension	128
	Embedding dimension	128
	Number of attention heads	4
	Bidirectional	true

Table 6: Attribute Extractor Model Parameters

Metric	Dual Bi-LSTM + CRF	Bi-LSTM + CRF	Transformer + CRF
F1 Score	**0.9329**	0.9308	0.9037
F1 Score (with domain)	**0.9311**	0.9289	0.8976
Precision	**0.9392**	0.9333	0.9132
Recall	0.9281	**0.9307**	0.9078
Model Size (MB)	6.4	**5.4**	27
Latency (ms)	77	**50**	116

Table 7: Attribute Extractor Results on our collected dataset

The results of our models on CoNLL2003 test set are given in Table 8. We see that our proposed model achieves reasonably high F1 score (within $\sim 1\%$ of current state of the art). A significant advantage of our approach is the simplicity of the model which allows it to be deployed on-device as well. Unlike our proposed model that involves training embeddings from scratch, all models that outperform our proposed model make use of powerful pre-trained or contextualized embeddings. Another interesting trend we observe is the similar pattern of performance of our three models across both datasets. This verifies that our proposed model performs the best for on-device entity extraction.

Models	F1 Score
LSTM-CRF (Lample et al., 2016)	90.94
Bi-LSTM-CNN-CRF (Ma and Hovy, 2016)	91.22
LM-LSTM-CRF (Liu et al., 2017)	91.25
Transformer-CRF (Ours)	91.75
Bi-LSTM-CRF + ELMO (Peters et al., 2018)	92.2
TENER (Yan et al., 2019)	92.63
BERT (Devlin et al., 2019)	92.8
Bi-LSTM-CRF (Ours)	92.85
Flair (Akbik et al., 2018)	93.1
Dual Bi-LSTM-CRF (Ours)	93.28
Cross Weigh + Pooled Flair (Wang et al., 2019)	93.43
Baevski et al. (Baevski et al., 2019)	93.5
LUKE (Yamada et al., 2020)	94.3

Table 8: Benchmark results on CONLL-2003 dataset

5.4 Entity Linker

For the entity linker we achieve an F1 score of 0.98. This module has a very high F1 score because most of the entity linking is done using string-matching and phonetic algorithms and regex pattern matching as compared to earlier modules, which involved machine learning/ deep learning models. The model size for this module is 284KB and latency is 40 ms;

5.5 Engine Pipeline

The complete engine pipeline is implemented as a service on device. The attribute extractor is implemented in PyTorch and the trained model is converted to android (v10) compatible version using the Pytorch JIT module. On-device inference is done using PyTorch android runtime. Similarly, Domain classifier is implemented in tensorflow and on-device inference is done using tensorflow-lite android library. Trained models were tested and deployed on Samsung Galaxy S10 and Note 10 devices.

The final end to end pipeline consists of 4 different modules, Classifier, Attribute Extractor, Entity Linker & Triplet Builder. The respective F1 scores achieved on the test set for the 4 modules are 95.96, 93.29, 98 & 92.4 respectively. The end to end accuracy of the overall system is 81.06 %. The outputs of each of these individual modules is included in the appendix. After porting to the device the model and app sizes were recorded to be 8.31 MB & 48.87 MB respectively, and engine's latency was measured to be ~200 ms.

6 Conclusion

In this paper, we have proposed a unique on-device pipeline to extract relevant information from unstructured sources such as messages, notification, calendar entries etc. By making on-device extraction very efficient, we eliminate the issue of user privacy while providing a platform for an enhanced personalised experience. Since the relevant domains are majorly non-overlapping, a Naïve Bayes classifier gives sufficiently good performance. For attribute extraction, we propose a dual word and character based Bi-LSTM + CRF model, which achieves best results on our self-curated test set as well as CONLL-2003 test set.

The feasibility of such a system was claimed through an on-device implementation using the proposed approach. The applications of such an on-device system can be envisioned across various personalization and recommendation services while maintaining user privacy.

7 Future Work

A possible extension of this work is to extend the English information extraction system to a multilingual one. This is an interesting area of exploration because each language has a different morphology, so it will be more challenging for a single model to capture multilingual features. Currently, our system is a pipeline consisting of several models, which can cause propagation of error. So, exploring the possibility of an end-to-end information extraction system is another direction in which we can expand our research.

References

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Tiago A. Almeida, José María G. Hidalgo, and Akebo Yamakami. 2011. Contributions to the study of sms spam filtering: New collection and results. In *Proceedings of the 11th ACM Symposium on Document Engineering*, DocEng '11, page 259–262, New York, NY, USA. Association for Computing Machinery.

Alexei Baevski, Sergey Edunov, Yinhan Liu, Luke Zettlemoyer, and Michael Auli. 2019. Cloze-driven pretraining of self-attention networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5360–5369, Hong Kong, China. Association for Computational Linguistics.

Richard Cooper, Sajjad Ali, and Chenlan Bi. 2005. Extracting information from short messages. In *Natural Language Processing and Information Systems*, pages 388–391, Berlin, Heidelberg. Springer Berlin Heidelberg.

Gordon V. Cormack, José María Gómez Hidalgo, and Enrique Puertas Sánz. 2007. Spam filtering for short messages. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, page 313–320, New York, NY, USA. Association for Computing Machinery.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Fatia Kusuma Dewi, Mgs. M. Rizqi Fadhlurrahman, Mohamad Dwiyan Rahmanianto, and Rahmad Mahendra. 2017. Multiclass sms message categorization: Beyond spam binary classification. In *2017 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, pages 210–215.

Tobias Ek, Camilla Kirkegaard, Håkan Jonsson, and Pierre Nugues. 2011. Named entity recognition for short text messages. *Procedia - Social and Behavioral Sciences*, 27:178–187. Computational Linguistics and Related Fields.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9:1735–80.

Huixing Jiang, Xiaojie Wang, and Jilei Tian. 2010. Second-order hmm for event extraction from short message. In *Proceedings of the Natural Language Processing and Information Systems, and 15th International Conference on Applications of Natural Language to Information Systems*, NLDB'10, page 149–156, Berlin, Heidelberg. Springer-Verlag.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, page 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016.

Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Yuanchun Li, Ziyue Yang, Yao Guo, Xiangqun Chen, Yuvraj Agarwal, and Jason I. Hong. 2018. Automated extraction of personal knowledge from smartphone push notifications. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 733–742.

Liyuan Liu, Jingbo Shang, Frank F. Xu, Xiang Ren, Huan Gui, Jian Peng, and Jiawei Han. 2017. Empower sequence labeling with task-aware neural language model. *CoRR*, abs/1709.04109.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Joseph Polifroni, Imre Kiss, and Mark Adler. 2010. Bootstrapping named entity extraction for the creation of mobile services. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

S. Vatsal, N. Purre, S. Moharana, G. Ramena, and D. Mohanty. 2020. On-device information extraction from sms using hybrid hierarchical classification. In *2020 IEEE 14th International Conference on Semantic Computing (ICSC)*, pages 178–181, Los Alamitos, CA, USA. IEEE Computer Society.

Zihan Wang, Jingbo Shang, Liyuan Liu, Lihao Lu, Jiacheng Liu, and Jiawei Han. 2019. CrossWeigh: Training named entity tagger from imperfect annotations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5154–5163, Hong Kong, China. Association for Computational Linguistics.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6442–6454, Online. Association for Computational Linguistics.

Hang Yan, Bocao Deng, Xiaonan Li, and Xipeng Qiu. 2019. Tener: Adapting transformer encoder for named entity recognition.