International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 05 | May 2020 www.irjet.net p-ISSN: 2395-0072

On Device Deep Neural Networks For Emoji And Reply Prediction

Steffan Mathew

Student, Dept. of Dual Degree Computer Applications, Sree Narayana Guru Institute of Science and Technology N.Paravur, Kerala,India

---***---

Abstract- *In this paper we investigate a architecture for training neural networks for prediction of emoji and text reply, so we take the best emoji prediction model out there: DeepMoji which is proven to provide state-of-the art accuracy. Millions of readily available emoji occurrences on Twitter were used to pretrain the model to learn a rich emotional representation. This knowledge was then moved tasks using a new layer-wise fine-tuning method to the target, obtaining improvements over the state-of-the art within a range of tasks: emotion, sarcasm and sentiment detection. For smart reply there is a neural network that decides whether or not to suggest responses, it's for not showing suggestions when they are unlikely to be used and keeps the system scalable LSTM neural network processes an incoming message, then uses it to predict the most likely responses. It can be improved by finding only the approximate best responses. To choose a small set of the most likely responses to show to the user that maximize the total utility we select responses from response space which is generated offline using a semi-supervised graph learning approach to deliver high response quality.*

Key Words: NLP-Natural Language Processing, LSTM-Long Short Term Memory, CPU-Central Processing Unit, GPU-Graphics Processing Unit, ML-Machine Learning

1.INTRODUCTION

In here we present a contemporary chat system, that aims to provide an easy-to-use and efficient means of communication along with a handful of useful services. The system is aimed to be a sufficient environment for social purpose and entertainment. One's ability to communicate can spell the difference between successor failure in all aspects of living. That's why there is a growing need for more tools and platforms that can be used to get the world together. Sentiments play a crucial role in deciding the meaning of our speech. It's not just the words that make sense, but the way they are said and the expressions that accompany them. Machine learning has made it possible for computers to imitate more and more human activities. A distinctive human activity is the ability to make of what others say and construct reasonable responses to that.

Deep neural networks are used here which are typically large, has so many parameters contained in it so training these network costs a high computing power with several CPUs and GPUs. To solve this problem we train small models for on-device prediction tasks, but this can lead to a drop in accuracy of the prediction limits the use of such models so by doing vocabulary pruning technique commonly applies to limit the parameters in model which yields lower memory footprint. Which inspired the learning of efficient on device ML models with low memory foots that can be run directly on device at low computation cost.

2.EMOJI PREDICTION

2.1 Work Flow

2.1.1 Pretraining

Emoji prediction model used data from Twitter from January 1st 2013 to June 1st 2017. Without URLs English tweets were used for the pretraining dataset, because the content obtained from the URL is important for understanding the emotional content.

All tweets were tokenized on a word-by-word basis. Words with 2 or more repeated characters were shortened to the same token, e.g. lol and loool. A special token was used for all URLs (only relevant for benchmark datasets), user mentions (e.g. '@acl2017' and '@emnlp2017'were thus treated the same) and numbers.

For multiple repetitions of the same emoji or multiple different emojis: for each unique emoji type, a separate tweet was saved for the pretraining with that emoji type as the label, regardless of the number of occurrences of each emoji.

The pretraining data was split into a training, validation and test set, where the validation and test sets were randomly sampled in such a way that each emoji was equally represented.

2.1.2 Model Description

Prediction model uses a variant of the Long Short-Term Memory (LSTM) model that has been successful at many NLP tasks. It uses an embedding layer of 256 dimensions to project each word into a vector space. A hyperbolic tangent activation function is used to enforce a constraint of each embedding dimension being within [-1, 1]. To capture the context of each word the model uses two bidirectional LSTM layers with 1024 hidden units in each (512 in each direction). Finally, By using skip connection all these layers taken by the attention layer is served as input. Fig-1 illustrates the model's architecture.

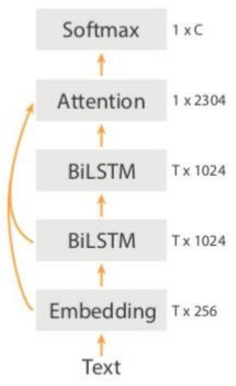


Fig -1: Illustration of the DeepMoji model with T being text length and C the number of classes.

The model decides the importance of each word for the prediction task by weighing them when constructing the representation of the text. A simple approach is used:

$$e_t = h_t w_a$$

$$a_t = \frac{exp(e_t)}{\sum_{i=1}^{T} exp(e_i)}$$

$$v = \sum_{i=1}^{T} a_i h_i$$

h_t : the representation of the word at time step t.
w_a: the weight matrix for the attention layer.
a_t : the attention importance scores for each time step.
v : the representation vector for the text.

This representation vector obtained from the attention layer v is a high-level encoding of the entire text, which is used as input to the final Softmax layer for classification.

2.1.3 Transfer Learning

The model was fine-tuned to the target task with freezing layers by disabling parameters updates to prevent overfitting. The model used a new simple transfer learning approach, presented by DeepMoji as chain-thaw, that sequentially unfreezes and fine-tunes a single layer at a time. It increases accuracy on the target task at the expense of extra computational power needed for the fine-tuning. By training each layer separately, the model is able to adjust the individual patterns across the network with a reduced risk of over fitting. Using the chain-thaw approach, each layer was fine-tuned individually starting from the first layer in the network.

The entire model was then trained with all layers. Each time the model converged as measured on the validation set, the weights were reloaded to the best setting, thereby preventing overfitting. Fig-1.1 illustrates this approach

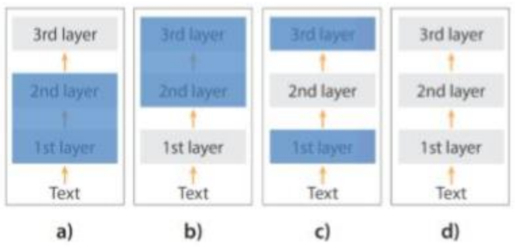


Fig -1.1: Iteratively training part of the network (starting with step a and finishing with step d). Blue layers are frozen and thus not uploaded.

2.2 Dataset

The dataset used was extracted from Twitter. Without URLs English tweets were used for the pretraining dataset, because the content obtained from the URL is important for understanding the emotional content.

A dataset of 56.6 billion tweets were used, which was then filtered to 1.2 billion relevant tweets. In the pretraining dataset, a copy of a single tweet was stored once for each unique emoji, resulting in a dataset consisting of 1.6 billion tweets. For valuate performance on the a validation set, pre training task and a test set both containing 640K tweets (10K of each emoji type) were used, the remaining tweets were used for the training set. Fig-1.2 shows the distribution of tweets across different emoji types.

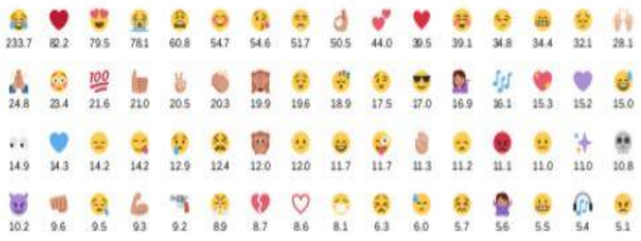


Fig -1.2: The number of tweets in the pretraining dataset associated with each emoji in millions.

3.SMART REPLY

3.1 Workflow

Smart reply consists of the following components (see fig - 2):

1.**Triggering model:** A neural network that decides whether or not to suggest responses. This improves utility by not showing suggestions when they are unlikely to be used and keeps the system scalable.

2.**Response selection:** An LSTM neural network processes an incoming message, then uses it to predict the most likely responses. It can be improved by finding only the approximate best responses
3.**Diversity:** To choose a small set of the most likely responses to show to the user that maximize the total utility.
4.**Response set generation:** Select responses from response space which is generated offline using a semi-supervised graph learning approach to deliver high response quality.

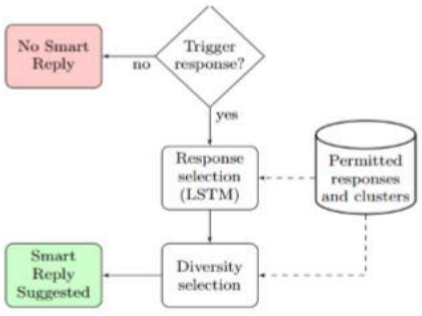


Fig -2: Smart Reply Components

3.1.1 Response selection

The set of all possible responses R and Given original message o, we would like to find:

$$r^* = \underset{r \in R}{\operatorname{argmax}} P(r|o)$$

The model itself is an LSTM whose input is the tokens of the original message

$$\{o_1, ..., o_n\}$$

and the output is the conditional probability of the sequence of response tokens given the input:

$$P(r_1, ..., r_m | o_1, ..., o_n)$$

This can be factorized as:

$$P(r_1, ..., r_m | o_1, ..., o_n) = \prod_{i=1}^{m} P(r_i | o_1, ..., o_n, r_1, ..., r_{i-1})$$

First, the sequence of original message tokens are read in, such that the LSTM's hidden state encodes a vector representation of the whole message. Then, given this hidden state, a soft max output is computed and interpreted as P (r1|o1,...,on) which is the probability for the first response token, as response tokens are fed in, the soft max at each timestep t is interpreted as P (rt|o1,...,on). then these soft maxes can be used to compute.

$$P(r_1, ..., r_m | o_1, ..., o_n)$$

Training: We want to maximize the log probability of observed responses, given their respective originals using stochastic gradient descent from given large corpus of messages:

$$\sum_{(o,r)} log P(r_1, ..., r_m | o_1, ..., o_n)$$

Both our input and output vocabularies consist of the most frequent English words in our training data after preprocessing.

Inference: We feed in an original message and then use the output of the soft maxes to get a probability distribution over the vocabulary. These distributions can be used in a variety of ways:
• From the response distribution draw a random sample
• By greedily taking the most likely token at each time step and feeding it back in we can approximate the most likely response.

3.1.2 Response set generation

We first need to define a target response space that comprises high quality messages which can be surfaced as suggestions. The goal here is to generate a structured response set that effectively captures various intents conveyed by people in natural language conversations. Target response space should capture both variability in language and intents.

• **Converting email responses:** The first step is to automatically generate a set of canonical responses messages that capture the variability in language. For example, responses such as "Thanks for your beautiful gift.", "Thank you for the gift!" convey the same information. We use its syntactic structure to generate a canonicalized representation and parse each sentence using a dependency parser.

• **Semantic intent clustering:** In next step, a cluster represents a meaningful response intent when we partition all response messages into "semantic" clusters (for example, "thank you" type of response versus "sorry"). We need to access semantic intents with its corresponding large corpus of sentences. However, it's neither at this scale nor readily available for our task. Moreover, unlike typical machine learning classification tasks, the semantic intent space can't be fully defined a priori. So instead, we model the task as a semi-supervised machine learning problem to automatically learn this information from data and a few human-provided examples.

• **Semi-supervised learning:** Next, From the manually labeled examples through the graph we learn a semantic labeling for all response nodes by propagating semantic intent information. We minimize the following objective function for response nodes in the graph.

• **Cluster Validation:** Finally, by sorting their label scores we extract the top k members for each semantic cluster. Then human raters validate the set of (cluster label, response) pairs. The raters are provided with a response Ri, a corresponding cluster label C (e.g., thanks) as well as few example responses belonging to the cluster (e.g., "HI!", "Hello.") and asked whether C belonged by Ri. The result is a validated set of high-quality response messages labeled with semantic intent and automatically. This is subsequently used by the response scoring model to search for approximate best responses to an incoming email and further to enforce diversity among the top responses chosen.

3.1.3 Suggestion diversity

To show the user we choose a small amount of option. A straightforward approach would be to just choose the N top responses and present them to the user. But these responses may be very similar. When the response options are not redundant at least one response being useful is greatest. To select a more varied set of suggestions we use diversity component using two strategies: omitting redundant responses and enforcing negative or positive responses.

• **Omitting Redundant Responses:** The user should never see two responses of the same intent. An intent is a cluster of responses that have a common purpose. In Smart Reply, there is exactly only one intent for every target response which is associated with it. The actual diversity strategy is simple which is the top responses are iterated over in the order of decreasing score. If its intent is already covered by a response on the suggestion list then it is added to the suggestion list. The resulting list contains only the highest-scored representative of each intent, and these representatives are ordered by decreasing score.

• **Enforcing Negatives and Positives:** The LSTM has a strong tendency towards producing positive responses, whereas negative responses such as I can't make it or I don't think so typically receive low scores. That is because positive replies may be more common. But it may be important to offer negative suggestions in order to give the user a real choice and this can be implemented through the following strategy:

–If none of the top three responses are negative, the third response is replaced with a negative one and the top two responses (after omitting redundant responses) contain at least one positive response
– In order to find the negative response, a second LSTM pass is performed. In this second pass, in the target set the search is restricted to only the negative responses.

3.2 Challenges

3.2.1 Response Quality

A high quality response is not necessarily the most probable. A response that happens frequently in our corpus may not be appropriate to surface back to users. For example, it could contain poor grammar or spelling (your the best!). Also it may be informal or politically incorrect.
So we use semi-supervised learning to construct a target response space R comprising only high quality responses.

3.2.2 Utility

Suggestions are most useful when they are highly specific to the original message, for example if the response is "Yes, I'll be there", it is common but unspecific response. So to improve specificity of responses, we apply some light normalization that penalizes responses which are applicable to a broad range of incoming messages. For example, the very generic "Yes!" has fallen out of the top ten. Also to increase the breadth of options shown to the user, we enforce diversity by exploiting the semantic structure of R. Passing each message through a triggering model we can improve the utility of suggestions that determines whether suggestions should be generated at all.

3.3 Dataset

• **Training set:** It is separated into 3 columns: the context of the conversation, the candidate response, and a flag or 'label' (= 0 or 1) denoting whether the response is a 'true response' to the context (flag = 1), or a randomly drawn response from elsewhere within the dataset (flag = 0). Train.csv is 463Mb, with 1,000,000 lines (ie. examples, which corresponds to 449,070 dialogues) and with a vocabulary size of 1,344,620.

• **Validation set:** Contains the validation set. Each row represents a question. Separated into 11 columns: the context, truth response or 'ground truth utterance', and 9 false responses or 'distractors' that were randomly sampled from elsewhere within the dataset. if it select the ground truth your model gets a question

utterance from amongst the 10 possible responses. valid.csv is 26.99Mb, with 19,551 lines and a vocabulary size of 115,687.

• **Testing set:**
Contains the test set. Formatted in the same way as the validation set. test.csv is 26.99Mb, with 18,920 lines and a vocabulary size of 115,622.

3.4 Output

Output of the model is mainly a list containing the generated responses by the model as indicated in the following example:
Input:
"How are you?"
Output:
• "I am fine, thank you"
• "Okay and you?"
• "Fine, thank you"
• "I am great"

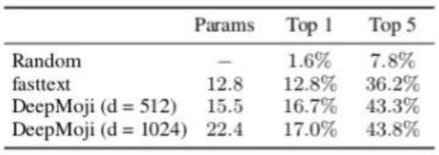
4.PERFORMANCE

4.1 Emoji Prediction

emoji prediction model is based on the DeepMoji model that was built and trained by MIT. It's by far the best model out there which is proven to provide state-of-the-art performance. In the following subsections we evaluate and analyze the model from various perspectives.

4.1.1 Experiments

The performance of the model is evaluated on the pretraining task with the results shown in figure-3 Both top 1 and top 5 accuracy are used for the evaluation with multiple emojis being potentially correct for any given sentence as the emoji labels are noisy. For comparison we also use a version of the model with smaller LSTM layers and a bag-of-words (fast text) classifier. For the latter classifier 256 dimensions were used, thereby making it almost identical to only using the embedding layer from our model.



	Params	Top 1	Top 5
Random	–	1.6%	7.8%
fasttext	12.8	12.8%	36.2%
DeepMoji (d = 512)	15.5	16.7%	43.3%
DeepMoji (d = 1024)	22.4	17.0%	43.8%

Fig -3: Accuracy of classifiers on the emoji prediction task d refers to the dimensionality of each LSTM layer. Paramaters are in millions.

The difference in top 5 accuracy between the fast text classifier (36.2%) and the largest model (43.8%) underlines the difficulty of the emoji prediction task. As the two classifiers only differ in that our model has LSTM layers and an attention layer between the embedding and Soft max layer, this difference in accuracy demonstrates the importance of capturing the context of each word.

4.1.2 Model Analysis

Many of the emojis carry similar emotional content, but have subtle differences in usage that the model is able to capture. Through hierarchical clustering on the correlation matrix of the model's predictions on the test set we can see that the model captures many similarities that one would intuitively expect (see Figure 3.1).

Figure 3.1 shows hierarchical clustering of the model's predictions across categories on the test set. It shows how the model learns to group emojis into overall categories and subcategories based on emotional content. The y-axis is the distance on the correlation matrix of the model's predictions measured using average linkage. Table 1 shows the description of benchmark datasets.

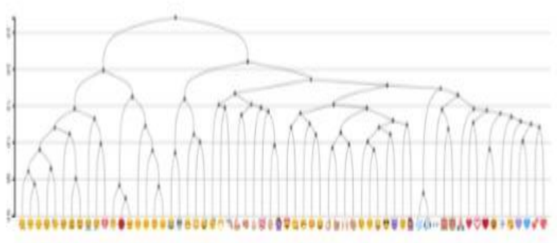


Fig -3.1: Hierarchical clustering of the model's predictions across categories

4.1.3 Benchmarking

DeepMoji was benchmarked on 3 different NLP tasks using 8 datasets across 5 domains. An averaged F1-measure across classes is used for evaluation in emotion analysis and sarcasm detection as these consist of unbalanced datasets while sentiment datasets are evaluated using accuracy. Table 1 shows the description of datasets used. For more information about these datasets please refer to DeepMoji's official illustration.

Table -1: Description of benchmark datasets

Identifier	Task	Domain	Clas ses	N_{train}	N_{test}
SE0714	Emotion	Headlines	3	250	1000
Olympic	Emotion	Tweets	4	250	709
PsychExp	Emotion	Experiences	7	1000	6480
SS-Twitter	Sentiment	Tweets	2	1000	1113
SS-Youtube	Sentiment	Video Comments	2	1000	1142
SE1604	Sentiment	Tweets	2	7155	31986
SCv1	Sarcasm	Debate Forums	2	1000	995
SCv2-GEN	Sarcasm	Debate Formus	2	1000	2260

Deep Moji model out performs the state of the art across all benchmark datasets. The new chain-thaw approach consistently yields the highest performance or the transfer learning. Results are averaged across 5 runs to reduce the variance. The statistical significance of the results were tested by comparing the performance of DeepMojivs. The state of the art. Bootstrap testing with 10000 samples is used. Results from DeepMoji are statistically significantly better than the state of the art with p < 0.001 on every benchmark dataset. Table 2 shows the results.

Table -2: Results across benchmark datasets

Dataset	Measure	State of the art	DeepMooji
SE0714	F1	.34	.37
Olympic	F1	.50	.51
PsychExp	F1	.45	.57
SS-Twitter	Acc	.82	.88
SS-Youtube	Acc	.86	.93
SE1604	Acc	.51	.58
SCv1	F1	.63	.69
SCv2-GEN	F1	.72	.75

4.2 Smart Reply

In this section, we describe the training and test data, as well as preprocessing steps used for all messages. Then, we evaluate different components of the Smart Reply system.

4.2.1 Data

To generate the training data for all Smart Reply models from sampled accounts, we extracted all pairs of an incoming message and the user's response to that message.

At the beginning of Smart Reply pipeline, data is preprocessed in the following way:

- **Language detection:** The language of the message is identified and non-English messages are discarded.
- **Tokenization:** message and subject body are broken into words and punctuation marks.
- **Sentence segmentation:** Sentences boundaries are identified in the message body.
- **Normalization:** Infrequent words and entities like personal names, URLs, email addresses, phone numbers etc. are replaced by special tokens.
- **Quotation removal:** Forwarded messages are removed and quoted original messages.
- **Salutation/close removal:** Salutations like Hi Mary and closes such as Best regards, John are removed. After the preprocessing steps, the size of the training set is 238.0 million messages, which include 153.0 million messages that have no response.

4.2.2 Results

Response selection results

Perplexity

Perplexity is a measure of how well the model has fit the data: a model which have lower perplexity assigns higher likelihood to the test responses, so we expect it to be better at predicting responses. Intuitively, when the model predicts the next word, there are on average k likely candidates means that a perplexity equal to k. In particular, we always know exactly what should be the next word, for the ideal scenario of perplexity equal to 1.

Table -3: Unique cluster/suggestions usage per day

	Daily Count	Seen	Used
Unique Clusters	376	97.1%	83.2%
Unique Suggestions	12.9k	78%	31.9%

Diversity results

We justify the need for a sizable response space R by reporting statistics around unique suggestions and both the diversity component and clusters in Table 3 Daily 12.9k unique suggestions are generated by the Smart Reply that belong to 376 unique semantic clusters. Out of those, people decide to use 4,115, or 31.9% of, unique suggestions and 313, or 83.2% of, unique clusters. However, that so many suggestions are never seened, as column 2 shows: the user may not open an email, use the web interface instead of mobile or just not scroll down to the bottom of the message. Also, only one of the three displayed suggestions will be selected by the user.

We observed, out of all suggestions used, 45% were from the 1st position, 35% from the 2nd position and 20% from the 3rd position. Since usually diverse responses is used by the third position, we conclude that the diversity component is crucial for the system quality.

Finally, we measured the impact of enforcing a diverse set of responses (e.g., by not showing two responses from the same semantic cluster) on user engagement: when we completely disabled the diversity component and simply suggested the three suggestions with the highest scores, the click-through rate decreased by roughly 7.5% relative. See figure 11.3.

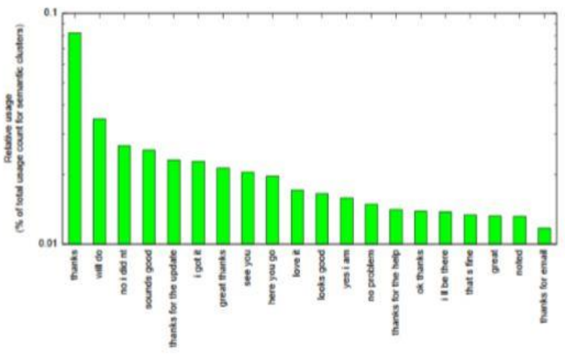


Fig-3.1: Usage distribution for semantic clusters corresponding to top suggested responses.

5.DISCUSSION AND FUTURE WORK

We intend to insert also a chat-bot which is an interactive agent or Artificial Conversational Entity that conducts a conversation via auditory or textual methods, it will use a sophisticated natural language processing system, or just scan for keywords within the input, then pull a reply with the most matching keywords, or the most similar wording pattern, from a database. The process of building a chat-bot can be divided into two main tasks: understanding the user's intent and producing the correct answer. The first task involves understanding the user input. In order to properly understand a user in put in a free text form, a Natural Language Processing Engine can be used. The second task may involve different approaches depending on the type of the response that the chat-bot will generate. Also, Response may be like Cleverbot's responses [www.cleverbot.com/] that are not pre-programmed. Instead, it learns from human input, as user type into his message and the system finds all keywords or an exact phrase matching the input. After searching through its saved conversations, it responds to the input by finding how the user responded to that input when it was asked. So our vision is to build such a chat-bot, that will interact with the user, respond to his questions, may about the history of his messages, friends, or any other statistics that may be useful for him.

6.REFERENCES

[1] WilliamFalcon.Googlesmartreply2017implementation intensorflow.https: //github.com/NextGenVest/google-smart-reply-2017. 2017.

[2] Bjarke Felbo et al. "Using millions of emoji occurrences to learn any domain representations for detecting sentiment,emotion and sarcasm". In:Conferenceon Empirical Methodsin Natural Language Processing(EMNLP) (Aug. 2017).

[3] Google Brain Team. TensorFlow open source machine learning framework. https://www.tensorflow.org/get_started (Nov. 2015).

[4] Anjuli KannanF et al. "Smart Reply: Automated Response Suggestion for Email". In: (Aug. 2016).

[5] Thomas Wolf. A pyTorch implementation of the DeepMoji model: state-ofthe-art deep learning model for analyzing sentiment, emotion, sarcasm etc. https://github.com/huggingface/torchMoji. 2017.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in Neural Information Processing Systems, pp. 1097–1105, 2012.

[7] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, et al., "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," IEEE Signal Processing Magazine, vol. 29, no. 6, pp. 82–97, 2012.

[8] Sequence to Sequence Learning with Neural Networks, Ilya Sutskever Google ilyasu@google.com, Oriol Vinyals Google vinyals@google.com, Quoc V. Le Google qvl@google.com

[9] Speech Recognition for Mobile Devices at Google, Mike Schuster, Google Research, 1600 Amphitheatre Pkwy., Mountain View, CA 94043, USA schuster@google.com

[10] Deep Neural Networks for Acoustic Modeling in Speech Recognition

[11] Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, vincent vanhoucke, Patrick Nguyen, Tara Sainath, and brian Kingsbury]

[12] Smart Reply: Automated Response Suggestion for Email

[13] "Android Wear 2.0: Make the most of every minute." https://blog.google/products/ android-wear/android-wear-20-make-most-every-minute

[14] B.-G. Chun and P. Maniatis, "Augmented smartphone applications through clone cloud execution," in Proceedings of the 12th Conference on Hot Topics in Operating Systems, HotOS'09, (Berkeley, CA, USA), pp. 8–8, USENIX Association, 2009.

[15] Deep Positron: A Deep Neural Network Using the Posit Number System

[16] Zachariah Carmichael§, Hamed F. Langroudi, Char Khazanov, Jeffrey Lillie, John L. Gustafson, Dhireesha Kudithipudi Neuromorphic AI Lab, Rochester Institute of Technology, NY, USA National University of Singapore, Singapore

[17] K. H. Lee and N. Verma, "A low-power processor with configurable embedded machine-learning accelerators for high-order and adaptive analysis of medical-sensor signals," IEEE Journal of Solid-State Circuits, vol. 48, pp. 1625–1637, July 2013.