Document Summarisation by key- sentence extraction using Neural Networks

Rudrajit Das Aditya Golatkar Akash Doshi Debarnab Mitra

Overview

- What is sentence extraction?
- Process for feature extraction
- Application of Fully Connected Neural Network with Ensemble Learning
- Application of Convolutional Neural Network
- Results achieved
- Problems and future work

What is sentence extraction?

Sentence extraction is a technique used for **automatic summarization of a text**. In this shallow approach, **statistical heuristics** are used to identify the most salient sentences of a text. Sentence extraction is a **low-cost approach** compared to more knowledge-intensive deeper approaches which require additional knowledge bases such as ontologies or linguistic knowledge. In short "sentence extraction" works as a **filter which allows only important sentences to pass**.

Our aim in this project is to extract key sentences from a document using **supervised learning**. We will explore and compare the performance of two popular forms of neural networks for this task - **feedforward neural networks with Ensemble learning** and **convolutional neural networks**.

Feature Extraction and Sentence Embedding

- We have considered a **corpus** of news articles given to us along with their associated summaries. Using the Natural Language Toolkit(**NLTK**) available in Python, we tokenized each article into its sentences. From each sentence, we then extracted the words

- We now used the **Google Dataset Word2Vec** to find the embedding vector for each word. This dataset takes into account all possible similarities between words, for example Mumbai and India have vector representations which are not too different. It represent words as **continuous vectors** in a low dimensional space and captures lexical and semantic properties of the words.

- These word vectors have to be linearly combined to give the sentence vector representation. The coefficients used for the linear combination are now described in the sentence embedding heuristic.

Heuristic for Sentence Embedding

- Compute the **weighted average of the word vectors** in the sentence and then **remove the projections** of the average vectors on their **first principal component** (“common component removal”).
- Here the weight of a word w is **a/(a + p(w))** with a being a parameter and p(w) the (estimated) word frequency.
- This method achieves **significantly better** performance than the **unweighted average** on a variety of textual similarity tasks, and on most of these tasks even beats some sophisticated supervised methods tested in (Wieting et al., 2016), including some RNN and LSTM models

Reference: Arora, Sanjeev, Yingyu Liang, and Tengyu Ma. "A simple but tough-to-beat baseline for sentence embedding's." (2016)

The theory behind this embedding

- The latent variable generative model treats corpus generation as a dynamic process, where the t^{th} word is produced at step t. The process is driven by the random walk of a discourse vector c_t.

- The discourse vector represents "what is being talked about." The inner product between the discourse vector c_t and the word vector v_w for word w captures the correlations between them. Then we have

$$\Pr[w \text{ emitted at time } t \mid c_t] \propto \exp\left(\langle c_t, v_w \rangle\right)$$

- To achieve a more realistic modelling, the probability of a word w is emitted in the sentence s is modelled by,

$$\Pr[w \text{ emitted in sentence } s \mid c_s] = \alpha p(w) + (1-\alpha)\frac{\exp\left(\langle \tilde{c}_s, v_w \rangle\right)}{Z_{\tilde{c}_s}},$$

$$\text{where } \tilde{c}_s = \beta c_0 + (1-\beta)c_s, \quad c_0 \perp c_s$$

The theory behind this embedding

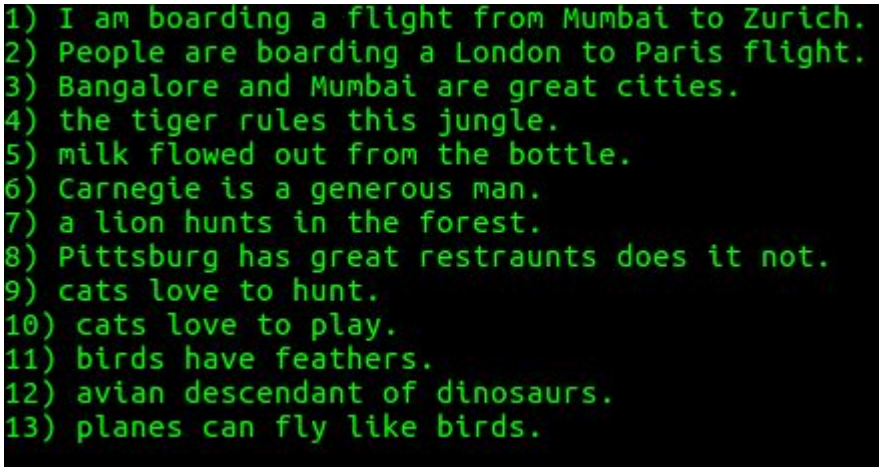
- Two terms have been introduced here. The first is an additive term **αp(w)** where **p(w)** is the unigram probability (in the entire corpus) of word and **α** is a scalar. This allows words to occur even if their vectors have very low inner products with $\mathbf{c}_s$.

- The second is a common discourse vector $\mathbf{c}_0$ which serves as a correction term for the most frequent discourse that is often related to syntax. It boosts the co-occurrence probability of words that have a high component along $\mathbf{c}_0$.

- The sentence embedding will be defined as the max likelihood estimate for the vector $\mathbf{c}_s$ that generated it. This can then be shown to generate the coefficients **a/(a + p(w))** where **a = (1 – α)/ αZ.** We used $a = 8 * 10^{-3}$

- To estimate $\mathbf{c}_s$, we estimate the direction $\mathbf{c}_0$ by computing the first principal component of $\tilde{\mathbf{c}}_s$**'s** for a set of sentences. In other words, the final sentence embedding is obtained by subtracting the projection of $\tilde{\mathbf{c}}_s$**'s** to their first principal component.

Algorithm in action



Similarity metric used is cosine similarity (normalized dot product)

Similarity vector for **6)** w/o using the prev. algo: [0.34246296, 0.31892182, 0.22637744, 0.27581882, 0.28438932, 1., **0.39794646**, 0.35118331, 0.24474546, 0.24870716, 0.11177552, 0.20496573, 0.10163028]

Similarity vector for **6)** using the prev. algo : [0.08545813, 0.03463596, 0.04396988, -0.03900207, 0.17517454, 1. , 0.09018801, **0.2086024** , 0.01897737, 0.04286888, -0.23284624, 0.03852853, -0.25666001]

Some more details...

- We now need to label each of the sentences as key or not-key sentence. For this we take each sentence in its summary, and compute its dot product with each sentence in the article(**Cosine Similarity**). The one with the max dot product is chosen as the key sentence.

- Having obtained a label for each sentence in the news article, we proceed to train it, using three different approaches: Fully Connected Neural Network, Convolutional Neural Network and Ensemble Learning(Weighted combination of the output of multiple neural networks)

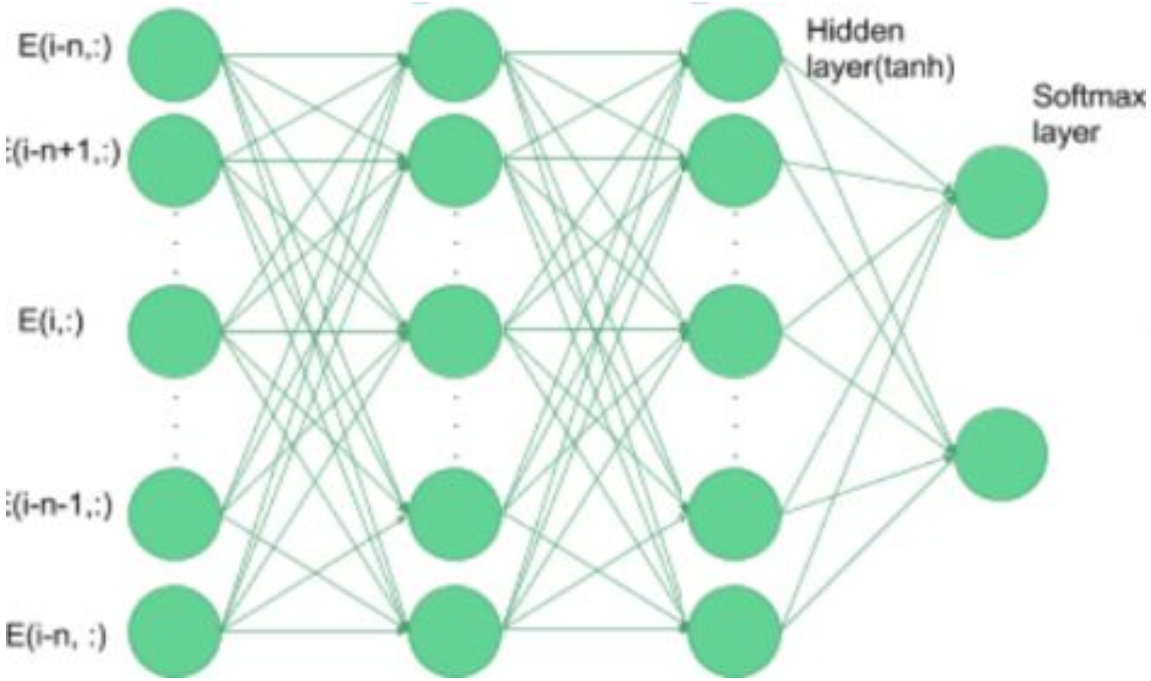
Fully Connected Neural Networks Implementation

- The embedding vectors of each sentence and its context window are fed in one at a time as input to the neural network. e.g. given a sentence S , we consider the surrounding sentences $\{S_{i-n}, S_{i-n+1}...S_{i-1},S_{i+1},..S_{i+n}\}$ as its context window.

- Also for each sentence S , we have constructed an embedding vector as explained before. For feedforward network, for each sentence, we concatenate and flatten $\{\mathbf{E_{i-n}}, \mathbf{..E_i} \mathbf{....}, \mathbf{E_{i+n}}\}$ as $F_i \in \mathbf{R^{(2n+1)m\times 1}}$ input feature which is a single column matrix.

- The label of the central sentence in the window is used as the training label for that window. So what we are predicting is $P(S_i \text{ is key sentence}| C_i)$ where C_i is the context window.

Fully Connected Neural Networks Implementation



- To avoid biasing the neural network to the not-key outcome(which is the vast majority), we are roughly training double the number of negative sentences as positive sentences. For this, we have randomly sampled from our overall dataset.

- The neural network contains two **tanh** layers and a final **sigmoid** layer. However, the data has not been normalized as that was found to adversely affect the performance.

“Bagging” of Fully connected Neural Networks

- Each NN in the ensemble is trained using a randomly drawn subset of the training set (with the ratio of negative to positive examples being nearly 2:1), and the average of the results of a **subset of these NNs** in the ensemble is taken as the final result.
- Seeking inspiration from (1), we choose those NN models for averaging whose validation accuracy exceeds a preset threshold λ. (The paper suggests including those NN’s whose weight exceeds a threshold λ, assuming that each neural network can be assigned a weight that could characterize the fitness of including this network in the ensemble.)
- In our case, all NN’s which have an accuracy of **> 0.5** (after tuning) were considered for averaging.
- With this architecture, we achieved an accuracy of 0.65 and an F1 score of 0.29.

(1) - Zhou, Zhi-Hua, Jianxin Wu, and Wei Tang. "Ensembling neural networks: many could be better than all." *Artificial intelligence* 137.1-2 (2002): 239-263

Results for Bagging based NN implementation

Obtained summary:

Cricketer-turned-politician Navjot Singh Sidhu, who recently joined the Congress, held a joint press conference with Captain Amarinder Singh on Thursday and said, "Baap baap hota hai, beta beta hota hai." Dismissing rumours that Sidhu is not on agreeable terms with Amarinder Singh, the Captain said, "I am his wicketkeeper. I will catch him in the slips."

Actual summary:

Setting aside all talks of alleged cold war, Captain Amarinder Singh and cricketer-turned-politician Navjot Singh Sidhu, who recently joined the Congress, held a joint press conference in Amritsar today.To this Captain said, "I am his wicketkeeper.I will catch him in the slips.

Results for Bagging based NN implementation

Obtained NN summary:

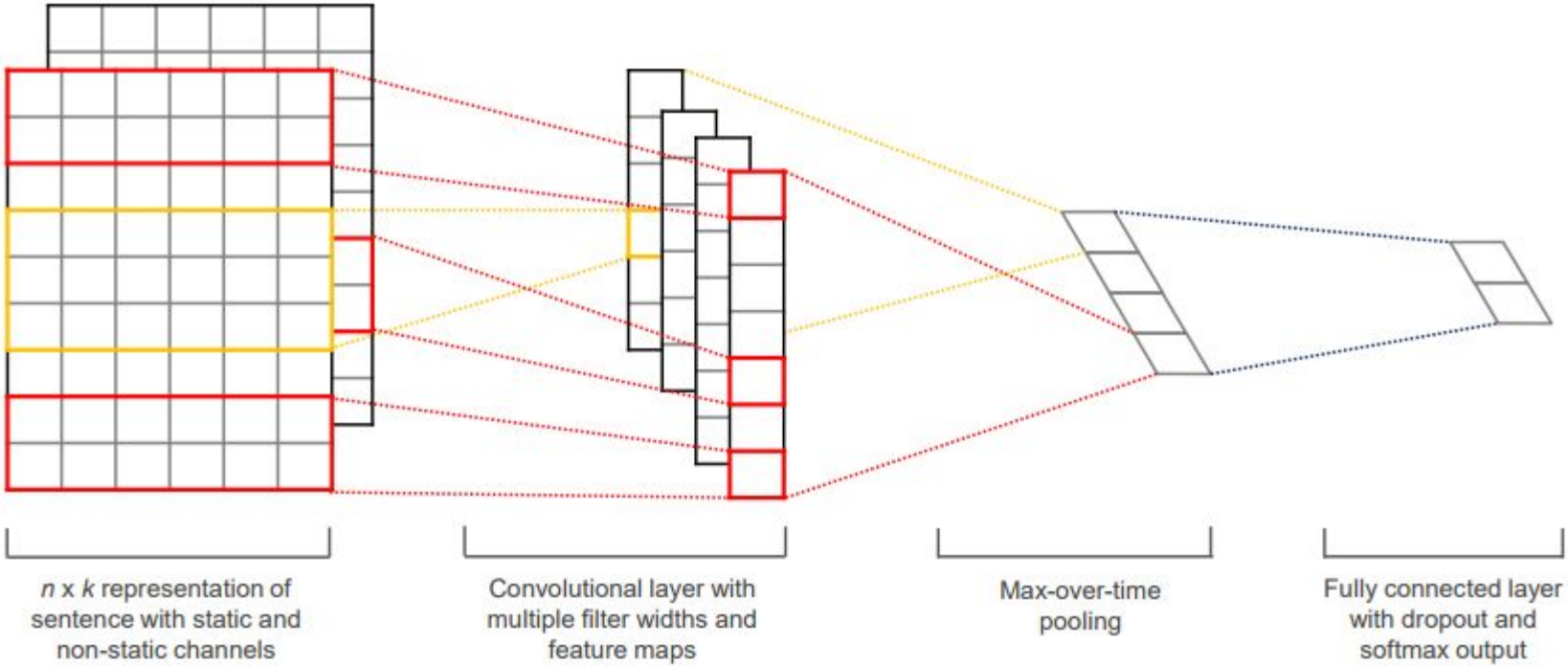
Global search engine giant Google has tied up with the Ministry of Consumer Affairs to raise awareness about online safety in India.Together, they will embark on a nationwide 'Digitally Safe Consumer' campaign, to help better protect consumer interest online, Google India announced on Saturday.Through the workshops, Google aims to provide people in India the desired training and necessary information on online safety tools.

Actual summary:

Google India has partnered with the Ministry of Consumer Affairs for a campaign called 'Digitally Safe Consumer' to help protect consumer interest online. The company has said it will provide training material to over 1,200 consumer organisations and consumer affairs departments of all Indian states. Google is also working with schools to spread awareness about internet safety among the youth.

Note: Third line in the Actual summary is not there in the document.

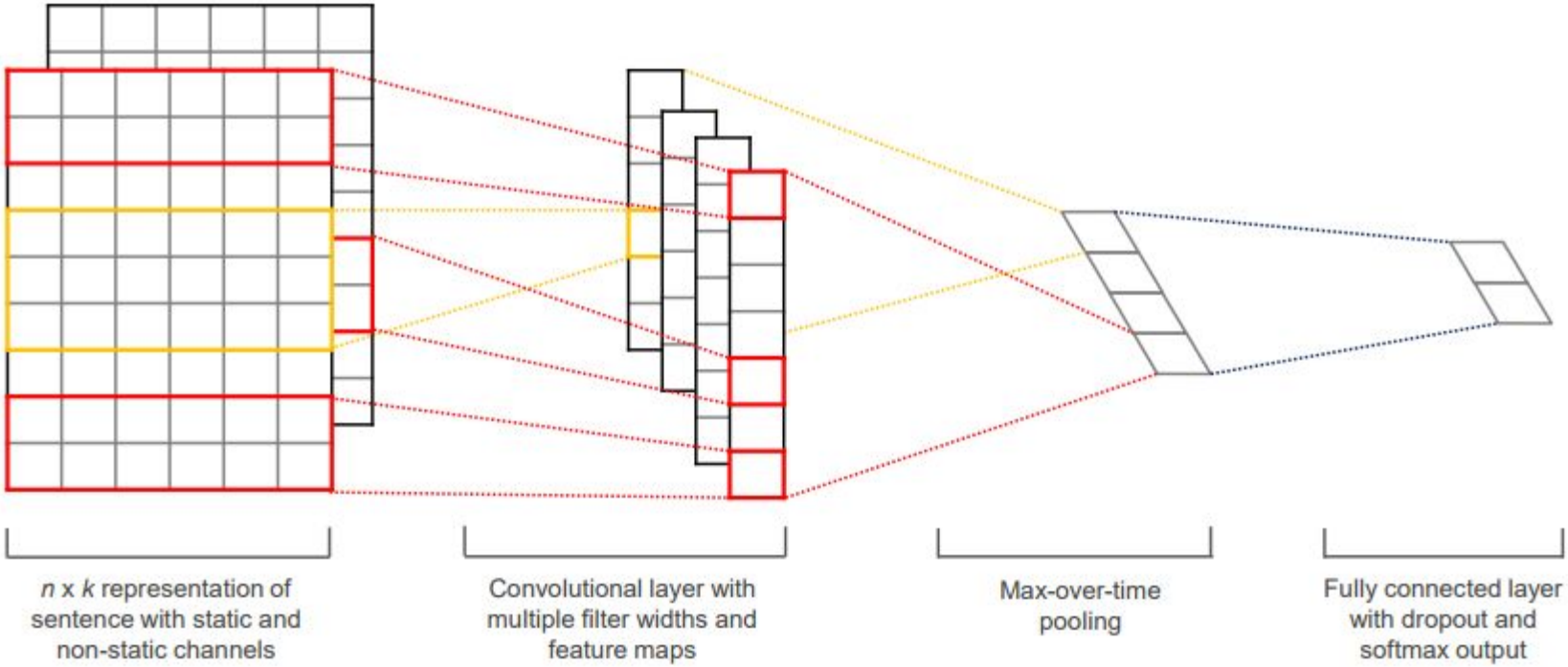
Convolutional Neural Network Implementation



[Yoon Kin, *Convolutional Neural Networks for Sentence Classification*]

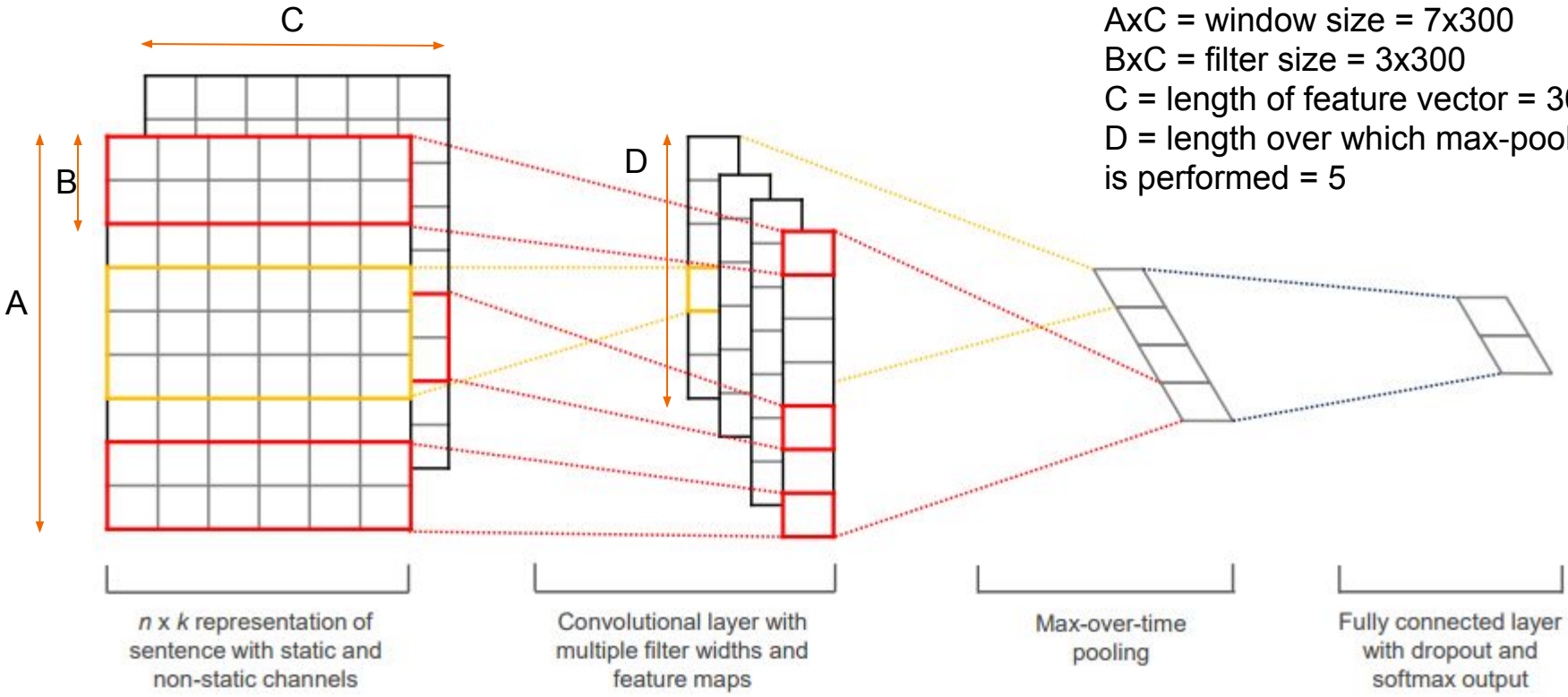
A feature $\mathbf{c}_i$ is generated from a window of words $\mathbf{x}_{i:i+h-1}$ by $\mathbf{c}_i = \mathbf{f}(\mathbf{w} \cdot \mathbf{x}_{i:i+h-1} + \mathbf{b})$. Here $\mathbf{b} \in \mathbf{R}$ is a bias term and $\mathbf{f}$ is a nonlinear function such as the hyperbolic tangent. This filter is applied to each possible window of words in the sentence $\{\mathbf{x}_{1:h}, \mathbf{x}_{2:h+1}, \ldots, \mathbf{x}_{n-h+1:n}\}$ to produce a feature map $\mathbf{c} = [\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_{n-h+1}]$ with $\mathbf{c} \in \mathbf{R}^{n-h+1}$

Convolutional Neural Network Implementation



We then apply a **max-pooling** operation over the feature map. The idea is to capture the most important feature—one with the highest value—for each feature map. These features form the penultimate layer and are passed to a fully connected layer which is finally connected to a **sigmoid** layer.

Convolutional Neural Network Implementation



AxC = window size = 7x300
BxC = filter size = 3x300
C = length of feature vector = 300
D = length over which max-pooling is performed = 5

Convolutional Neural Network Modification

- CNN architecture described previously did not yield very promising results.
- Instead, we treated the 300x7 matrix as an single channel image.
- We then used a convolutional layer with 128 filters and 3x3 kernel. We used “tanh” activation function.
- Next, we added a 2x2 pooling layer.
- We then flattened the output of the max-pooling layer and connected it to a 256x1 fully connected layer with “tanh” activation and then finally to a sigmoid neuron.
- For regularization (to prevent overfitting) we incorporated a dropout of 0.5.
- Using this modified architecture, we achieved an accuracy of 0.78 and an F1 score of 0.46.

Results for CNN Implementation

Obtained CNN Summary:

Canadian-born Indian YouTube sensation Lilly Singh has emerged as the Favourite YouTube Star 2017 at the 43rd People's Choice Awards!Lilly, who also goes by the name IISuperWomanII, was up against some stiff competition, with YouTubers PewDiePie, Shane Dawson, Tyler Oakley, and Miranda Sings, but walked away as 2017's favourite star.twitter.

Actual Summary:

Indo-Canadian YouTube personality Lilly Singh, also known as Superwoman, was named the Favourite YouTube Star at the 2017 People's Choice Awards in Los Angeles. PewDiePie, Miranda Sings, Shane Dawson and Tyler Oakley were the other nominees in the category. With over 10 million subscribers on YouTube, numerous celebrities including Priyanka Chopra have featured in Lilly's videos.nn

Note: Third line in the Actual summary is not there in the document.

Results for CNN Implementation

Obtained CNN Summary:

Superstar Rajinikanth on Friday extended his support for jallikattu, the popular and ancient bull-taming sport, played usually around Pongal festival in Tamil Nadu.He said it must be held as it is part of Tamil culture.Last year, the Supreme Court banned jallikattu, earning the wrath of its supporters and well-wishers."After Kamal Haasan's statement, several Kollywood celebs backed jallikattu, including Dhanush, Simbu, Khushbu Sundar, GV Prakash and RJ Balaji.

Actual Summary:

Superstar Rajinikanth on Friday extended his support for Jallikattu, the popular and ancient bull-taming sport, played during Pongal festival in Tamil Nadu. Rajinikanth said, "Bring in whatever rules but Jallikattu must be held to keep up the traditions of our Tamil culture." Earlier, Kamal Haasan had also supported Jallikattu, saying, "If you want a ban on Jallikattu, ban biriyani too."

Where CNN performed better than NN

Obtained NN summary:

Canadian-born Indian YouTube sensation Lilly Singh has emerged as the Favourite YouTube Star 2017 at the 43rd People's Choice Awards!"I am so grateful.

Obtained CNN Summary:

Canadian-born Indian YouTube sensation Lilly Singh has emerged as the Favourite YouTube Star 2017 at the 43rd People's Choice Awards!Lilly, who also goes by the name IISuperWomanII, was up against some stiff competition, with YouTubers PewDiePie, Shane Dawson, Tyler Oakley, and Miranda Sings, but walked away as 2017's favourite star.twitter.

Results for CNN Implementation
Better Summary obtained by our system

Obtained CNN Summary:

England pace bowler Jake Ball on Tuesday said the visitors will try to unsettle Indian skipper Virat Kohli with short balls and not let him find his rhythm while batting during the second One-Day International (ODI) cricket match.Kohli scored a gritty 122 for the hosts to better the visitors in the opening game of the three-match ODI series at Pune on Sunday, pulling them 1-0 ahead.The second match will be played here on January 19.

Actual Summary:

England fast bowler Jake Ball on Tuesday said that England bowlers will use short balls to unsettle Virat Kohli and will not let him find his rhythm while batting during the second ODI. "He's an unbelievable player. We've got plans for him and, hopefully, we can put them into practice in a couple of days' time," added Ball.

Problems

1. Finding feasible datasets were hard. Moreover there were lot of a spurious characters in the dataset, which caused difficulty in extracting the correct sentences.
2. More importantly, the computation power at our disposal was a severely limiting factor. It was taking very long to train more than 1500 documents, and this severely affected the accuracy on test articles while identifying the correct sentences.
3. There were many words in the Indian News Articles which did not have a word vector representation in the Google dataset. This affected the key sentences identified.
4. Some of the files in the datasets were difficult to calculate truncated SVD for due to various numerical errors.

Future Work

1. What we have done here is simply to extract the key sentences, we have not formed new sentences in the summary. However by doing a subject-object-predicate analysis, we can construct a dependency graph and each of the edges can be appropriately weighted to determine which predicate-object pairs should be picked for the given subject. This or some other form of abstractive text summarization can be employed.
2. We can improve the sentence embedding algorithm slightly by considering Taylor Series Expansion up to the second order term as it doesn't seem small enough to be neglected. We might also consider approximating the exponential term by a linear term as the inner product could be small enough to perform linear approximation (just to simplify things).

Contributions

Akash : Feature Extraction

Rudrajit : Neural Networks and Ensemble Learning

Aditya : Convolutional Neural Networks and Modifications

All three of us worked together and in sync.