Deep Q-Networks for Event Summarization

Francisco Javier Arceo, Chris Kedzie

Columbia University in the City of New York

December 13, 2016

Event Summarization

- In the extractive streaming summarization task, we are given as input a query (i.e., a short text description of a topic or an event), a document stream, and a time ordered set of sentences relevant to the query.
- Starting with an initially empty summary, an extractive, streaming summarization algorithm is intended to examine each sentence in order and, when new and important (relative to the query) information is identified, add that sentence to the summary.

Event Summarization

- Recent research in text retrieval has focused on extractive algorithms to identify important sentences from a large set of documents (e.g., [1], [4], [2], and [3]) for summarizing articles from different events in the news.
- [4] has shown that it is possible to select relevant sentences from a massive number of documents on the web to create summaries with meaningful content by adapting classifiers to maximize search policies.
- These systems have been shown to fall short of algorithms that employ simple heuristics [2], which may be due to inadequate capturing of the rich structure and often idiosyncratic information by traditional n-gram language models.

Event Summarization

This leads us to explore Deep Q-Networks (DQN) for 3 reasons:

- Both the representation and interaction between the stream, summary, and query can be learned
- The embeddings can learn a more robust semantic representations than classic n-gram models by using a RNN-LSTM
- By randomly exploring the state space, the ϵ-greedy strategy learns a policy that yields more consistency between train and test distributions

States

A state $s(x_t, \tilde{y}_t, d)$ at time t is a function of the stream X_d, $\forall d \in D$, which represents the candidate sentence, x_t to include in the summary, the state of the current summary $\tilde{y}_t$, and the query d. For brevity we use s_t where the dependence on $x_t, \tilde{y}_t$, and d is assumed.

Actions

We define an action, a_t, at each time step from our set of actions as $\mathcal{A} := \{select, skip\}$ where *select* corresponds to adding the current sentence x_t to the predicted summary $\tilde{y}_t$ and incrementing the current system time, or *skip* where only t is incremented without changing the current summary.

Reward

The reward r for a given action a at time t is measured by the change in ROUGE-NF1 score of the predicted summary $\tilde{y}_t$ measured against a gold standard summary Y. That is,

$$r_t = \text{ROUGE-NF1}(\tilde{y}_t, Y) - \text{ROUGE-NF1}(\tilde{y}_{t-1}, Y). \quad (1)$$

Policy

- A policy takes as input a state and a set of possible actions and returns the optimal state, this policy can be deterministic or probabilistic. We define our policy as a Q-Learner using both an LSTM and a bag-of-words model
- We define an architecture similar to that of [5] and map our three inputs (query, sentence, and current predicted summary) into LSTM embeddings according to **Figure 1**
- Our extraction policy then takes as input the state s_t at time t and returns the expected optimal action

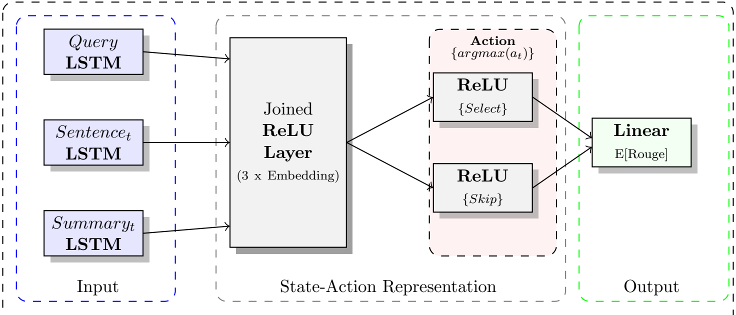

Figure 1: DQN-LSTM Architecture

DQN-LSTM for Event Summarization Training Procedure

Input: $\{\mathcal{D}$: Event queries, X_d: Input sentences, N: Number of epochs$\}$
Output: $\{\hat{Q}$: extraction policy, $\tilde{Y}_d$: event summary for query $d\}$

Initialize extraction policy $\hat{Q}$ with random weights
Initialize memory and summary: $\Gamma, \tilde{Y} = \{\emptyset\}_{d=1}^{|\mathcal{D}|}, \{\emptyset\}_{d=1}^{|\mathcal{D}|}$
for *epoch* $= 1, .., N$ **do**
    for query $d \in \mathcal{D}$ **do**
        $X_d, \tilde{Y}_d = \{\text{Extract } t = 1, ..., T_d \ (sentences_d, \ summary_d)\}$
        for $x_t, \tilde{y}_t \in X_d, \tilde{Y}_d$ **do**
            Set $s_t = s(x_t, \tilde{y}_t, d)$
            $\forall a_t \in \mathcal{A}(s_t)$ compute $\hat{Q}(s_t, a_t)$ and select $a_t^* = \text{argmax}_{a_t} \hat{Q}(s_t, a_t)$
            if $random() < \epsilon$ **then** select a_t^* at random with $\Pr(a_t) = \frac{1}{|\mathcal{A}|}$
            Update $\tilde{y}_{t+1}$ according to equation (1)
            Execute action a_t^* and observe reward r_t and new state s_{t+1}
            Update $\Gamma_d = \Gamma_d \cup \{[s_t, a_t^*, r_t, s_{t+1}]\}$
    for $j = 1, ..., J$ transitions sampled from Γ **do**

$$\text{Set } y_j = \begin{cases} r_j & \text{if } s_{j+1} \text{ is terminal} \\ r_j + \gamma \max_{a'} \hat{Q}(s_{j+1}, a'; \theta) & \text{if } s_{j+1} \text{ is non-terminal} \end{cases}$$

        Perform gradient step on $\mathcal{L}(\theta) = (y_j - \hat{Q}(s_j, a_j; \theta))^2$

We benchmark our model against (1) an oracle that greedily chooses to include sentences if the increase in ROUGUE-NF1 is positive, (2) random selection of sentences, and (3) a bag-of-words multilayer perceptron.

Useful Links below

Thank you

- GitHub

Fernando Diaz, Bhaskar Mitra, and Nick Craswell.
Query expansion with locally-trained word embeddings.
In *Proceedings of the 54th Annual Meeting of the Association for Computational*. ACL–Association for Computational Linguistics.

Cristina Gârbacea and Evangelos Kanoulas.
The university of amsterdam (ilps. uva) at trec 2015 temporal summarization track.

Chris Kedzie and Kathleen McKeown.
Extractive and abstractive event summarization over streaming web text.
In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 4002–4003, 2016.

Chris Kedzie, Kathleen McKeown, and Fernando Diaz.
Predicting salient updates for disaster summarization.
In *Proceedings of the 53rd annual meeting of the ACL and the 7th International Conference on Natural Language Processing*, pages 1608–1617, 2015.

Karthik Narasimhan, Tejas D Kulkarni, and Regina Barzilay.
Language understanding for textbased games using deep reinforcement learning.
In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Citeseer, 2015.