Introduction & Motivation

- In this work we propose a machine learning paradigm for costumer-care automation.
- Traditional costumer-care systems are usually handled by human agents, and there are drawbacks:
  - Long waiting time
  - Repeated questions
  - High business costs
- The automatic one, reduces the cost significantly, and shortens the waiting time.

Question/Answer Embedding

- The first step is to learn a mathematical representation of both questions and answers.
- Two major approaches:
  - Bag of word and its extensions.
  - Neural network based embedding.
- Here we used doc2vec approach, which is an extension of word2vec embedding for sentences and paragraphs.
- It starts by a 1-hot vector representation of the words (and paragraphs), and then learns an embedding, in a way that we can predict the current word given its context.
- Its block diagram is shown below:

    * Doc2vec block diagram, courtesy of Mikolov

        It learns these embedding by maximizing the log-likelihood of current word given its previous and future words as:

        \[ \sum_t \sum_k \log p(w_t | w_{t-k}, \ldots, w_{t+k}, s) \]

General Idea

- We treat this problem as a form of question-answering task in natural language processing.
- This approach has two major steps:
  - Question and answer embedding.
  - Learning the similarity of questions and answers.

Similarity Learning

- We use the knowledge embedded in the past data to learn how suitable an answer is for a given question.
- We propose a neural network which takes a pair of question and answer embedding as the input and predicts how similar they are.

Experiments

- Evaluation on the Insurance QA dataset, that contains a training set of 12,889 questions, a validation and a test sets of 2,000 questions, with a pool of 100 candidate answers.
- Batch size of 100, and train for 600 epochs.
- Accuracy

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag of word + SVM</td>
<td>0.72</td>
</tr>
<tr>
<td>The proposed algorithm</td>
<td>0.83</td>
</tr>
</tbody>
</table>

Accuracies v.s. epochs

Training and Test Accuracies