Modeling Long-Distance Dependencies and Modular Representations for Natural Language Processing
Summary

Despite progress in deep neural networks, NLP still faces a huge challenge in terms of taking into contextual information and modeling the same. According to, Anna Rumshisky, the author and presenter of the paper, believes that this is simply because there are few attempts, if any, to align computational models with how humans use language.

In this paper, they have attempted to address the functional specialization of language-processing regions in the brain as well as the long-distance dependencies in language. For the former, she has used a neural network architecture that uses adversarial training to learn such modular representation and attempts to dissociate meaning from form for linguistic input.Also, to take into account context a neural model that uses an updatable external memory component to capture contextual information has been employed.

Some of the key concepts from the talk were as follows:
- **Seq2seq models for generation** can be used for applications such as question-answering, machine translation and dialogue generation.
- **Recurrent Neural Networks (RNNs)** - handle variable length inputs/outputs, unrolled and trained with backpropagation to compute gradient.
- For the purposes of classification/regression in RNN architecture, input sequence is encoded into a single vector and for generation of output sequence is achieved using a single vector.

As part of her experiments, she and her team have tried to dissociate meaning from form. For example, there are differences in Shakespearean English and Modern English. However, it's possible to come to the same conclusions in terms of meaning even though the representations are different. A similar thing can be achieved, to understand sentiment polarization in texts through sentiment analysis.

To train the proposed DissoNet architecture, consisting of discriminator and generator, the following process was used:
- Two losses combined

    - Discriminator loss L(d)
    - Encoder-generator loss, L(EG) = L(rec) - L(d)
- Training procedure for each batch of the data:
    - Train the discriminator with L(d)
    - Train the encoder and the generator with L(EG)

As a result, they found that it is enough to have just the discriminator and the adversial loss to force the model to learn to dissociate the form and the meaning.