Adverse Drug Extraction in Twitter Data using Convolutional Neural Network

Liliya Akhtyamova, John Cardiff, Mikhail Alexandrov

ITT Dublin
Autonomous University of Barcelona

TIR Workshop 2017

Motivation

- **Adverse Drug Reactions** (ADR) ? unintended responses to a drug when it is used at recommended dosage levels

- **Side effects** of medicines lead to **300 thousand** deaths per year[1] in the USA and Europe

- **Patients** are not reporting side effects adequately through official channels

[1] Businaro R., Why We Need an Efficient and Careful Pharmacovigilance? Journal of pharmacovigilance, 2013

Motivation

Patients are actively involved in sharing and posting health-related information in various **healthcare social networks**:

- a large source of recent data from all over the world
- diverse information about the majority of drugs
- broad distribution of patients

Thus, can use this data to estimate ADRs

▷ Tremendous task to be performed manually

▷ Need an automated way of doing this

Processing of Drug-Related Posts on Twitter

The following **challenges** occur:

1. short posts formats
2. complexity of human language
3. unbalanced structure of data

In this work, we try to solve them by proposing:

- ▷ a CNN-based method for ADR classification

ADR Classification Dataset

ADR Dataset

Dataset:

- dataset obtained from the PSB 2016 Social Media Shared Task for ADR classification (Task 1)[2]
- 7,574 instances (about 10% are positive)
- information about over 100 drugs

Additional data source:

dataset for sentiment analysis classification task from Semeval-2015[3]

[2] http://diego.asu.edu/psb2016/task1data.html
[3] http://alt.qcri.org/semeval2015

ADR Dataset

Frequent misspellings: "Baek suddenly losing his glow :(nd im losing my abilify to speak"; "adderal reeeeeeallllllly helped my depression but I had terrible s/e's :(Do you have Hypothyroidism?"

Confused sentiment: "I loved effexor for anxiety and depression but it raised my blood pressure too much so I had to stop"

Drug abuse: "Sertraline Buspirone Lexapro and Abilify really messed up. I felt like Theon Greyjoy :("

Drug-drug interaction: "I'm in pain. I mixed my antibiotics with my lexapro, and now I feel like I have the flu. :("

Overall experience: "apparently itching/rash can be a side effect of wellbutrin that doesn't show up for a while after u start taking it? This is fine:("; "copaxone injections in the next week or so, got my health insurance sorted thankfully. Kinda nervous about the side effects"

Other bad sentiment: "not sure id be so brave with the heights! I'm not bad, struggling with appetite, pain and bloating :(may have to dbl humira."; "okay I only have 2 pain pills left :(no more lexapro , my knee hurts . :/"

Method

Problem Formulation

- Given an input text post T, the goal is to predict whether it mentions ADR or not R_T
- A CNN $\mathbf{F}_W$ parameterized by weights W is used to learn a decision function
- Given the training set $\{T_i, R_{T_i}\}_{i=1}^{N}$ consisting of N post-rating pairs, the CNN is trained to minimize cross-entropy loss function

Input Processing

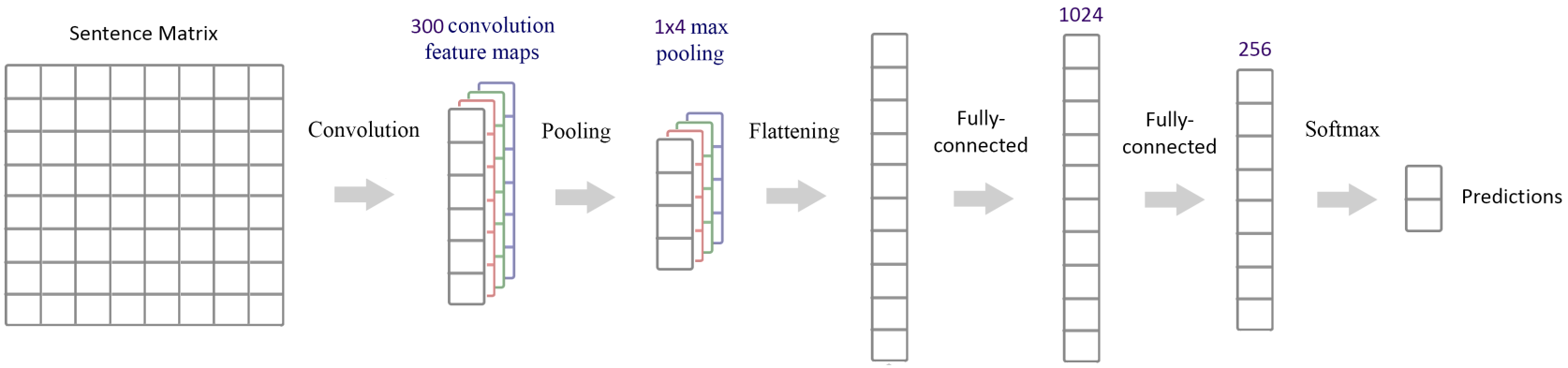
- **Input:** post $\mathbf{T}$ treated as an ordered sequence of words $\mathbf{T} = \{w_1, w_2, ..., w_N\}$
- **Plain words** are mapped to their vector representations using *word2vec*:
$$w_i \rightarrow \mathbf{w_i}$$
- ... and **stacked** together into a sentence matrix $\mathbf{M_T} = [\mathbf{w_1}, \mathbf{w_2}, ..., \mathbf{w_N}]$

$\rightarrow$ Matrix $\mathbf{M_T} \in \mathbb{R}^{D \times N}$ is used as an input data for our CNNs

- Additionally pretrained GoogleNews[4] and Wikipedia[5] word embeddings were used

[4] https://code.google.com/archive/p/word2vec/
[5] https://fasttext.cc/docs/en/english-vectors.html

General CNN Architecture



1. convolutional layer: 300 filters of size $5 \times D$
2. max-pooling layer
3. two fully-connected layers: 1024 and 256 neurons

Regularization: l_2-norm and *dropout*

Experiments

Technical Details

Word embeddings:

- context window size of 5
- words with frequency less than 5 are filtered
- dimensionality D of word embeddings – 300

Convolutional Neural Networks:

- trained for 20K iterations
- learning rate – 5e-4
- l2-regularization set to 0.01, dropout rate – 0.2

Methods

- **Bag-of-words model** – takes into account the multiplicity of the appearing words

  text $\rightarrow$ a vector with values indicating the number of occurrences of each vocabulary word in the text

  classification $\rightarrow$ Logistic Regression or Random Forest *(500 trees)*

- **Single CNN** – with own and pretrained word embeddings; with additional data source – sentiment data and without

Classification performances over the original and augmented data sets

Training data	Method	ADR F-score, %	Non-ADR F score, %	Accuracy, %
Huynh et al.	CNN+glove	0.51	-	-
original	bow+logistic regression	0.367	0.851	71.0
	CNN+word2vec	0.324	0.732	61.6
	CNN+word2vec(+2.5m)	0.426	0.892	81.6
	CNN+word2vec(+0.2m)	0.483	0.936	88.6
	CNN+GoogleNews	**0.542**	0.946	90.4
	CNN+Wikipedia	**0.540**	0.942	90.2
original +0.2m	CNN+word2vec	0.301	0.687	56.7
	CNN+word2vec(+2.5m)	0.373	0.914	87.5
	CNN+word2vec(+0.2m)	0.465	0.934	88.2

Discussion

Summary:

- end-to-end solution that is based on a CNN with pretrained GoogleNews word embeddings
- ability to handle with imbalanced structure of data
- computational experiments, demonstrating a strong advantage of the proposed solution over the standard approaches

Future Work:

- more intricate preprocessing
- building a committee of different models (e.g. ensemble, bagging or boosting)
- augmentation of the existing dataset with data from other healthcare networks (forums, specialized medical websites)