M.S. Project: Propaganda Detection

Mitchell P. Kristie III

December 2019

1 Introduction

For my Master's Project I am participating in SEMEVAL 2020 Task 11: Propaganda Detection. Participants are tasked with developing automatic methods for identifying propaganda techniques in text. Task 11 is divided into two sub-tasks: propaganda detection and propaganda classification. This project focuses on sub-task 1, detection. This task is a binary sequence classification task. For this task a conditional random field was created and trained on manually extracted features from the provided corpus. For this project I explored how various features and training parameters affected performance. Also I explore certain pitfalls impacting the classification and proposals going forward to improve the task. The competition ends mid-March.

2 Task Description

Sub-task 1 can be described as a binary classification task on sequences of strings. Given an articles of varying lengths, identify which spans of strings within the article are propaganda.

3 Corpus

The Propaganda Technique Corpus was provided by the SEMEVAL 2020 Task 11 staff. It consists of hand-labeled articles from 48 different news sources. The articles are formatted in that the first line consists of the article title, while subsequent lines make up the article itself. Each article had two companion files which provided information on the location of propaganda in the article (task 1) and what type of propaganda it was (task 2). Not all files contained propaganda techniques.

3.1 Corpus Statistics

Propaganda Technique Corpus (PTC)

Training and Testing Articles: 371
Total Lines: 17k
Total Tokens: 34k
Number of Propaganda Spans: 5468

3.2 18 Propaganda Techniques:

The competition staff identified 18 separate techniques and provide detailed descriptions of each. Spans in the corpus are not limited to single techniques and thus some spans may actually include several different instances of propaganda.

Loaded Language	Labeling	Repetition
Straw Men	Flag-Waving	Exaggeration
Doubt	Appeal to fear-prejudice	Reductio ad Hitlerum
Causal Oversimplification	Slogans	Appeal to Authority
Black-and-White Fallacy	Thought-terminating Cliches	Whataboutism
Red Herring	Bandwagon	Obfuscation

4 Machine Learning Models and Optimization

For sequence tagging I used a CRF and tested five different optimization algorithms. The implementation was written in python using sklearn-crfsuite.

Conditional Random Field:

> The model I used for sequence labeling was a conditional random field(Lafferty 2001). Conditional Random Fields are an extension of Hidden Markov Models that incorporate bi-directionality and model the conditional probability of a sequence of target labels given a sequence of inputs. CRF's are advantageous in that they help to avoid the label-bias problem. The label-bias problem occurs when states within the probabilistic graph have few or only one outgoing transition and thus effectively ignore their input (Lafferty 2001).

Optimization Algorithms:

- L-BFGS: Limited-memory BFGS (Nocedal 80)
- l2sgd: Stochastic Gradient Descent (Shalev-Shwartz 07) with l2 regularization
- ap: Averaged Perceptron (Collins 02)
- pa: Passive Aggressive (Crammer 06)
- arow: Adaptive Regularization Of Weight Vector (Mejer 10)

5 Feature Engineering

Each article went through a series of pre-processing steps to annotate it in a format suitable for the CRF model. This included sentence segmentation, tokenization and various forms of tagging. N-grams of different dimensions were created. Additional features including title information, case information, and stopword information were also included. I/O encoding (Jurafsky 2009) was used to classify tokens as propaganda or other. The two sentiment models offered different sets of features and were used at both the token and sentence level. Sentiment and title information were specifically included to try to address the nature of some of the propaganda techniques, while the more general features were incorporated in hopes that they would capture general syntactic constructions.

N-Grams	Sentiment Analysis	POS	Token Level	Class Encoding
Unigram	Flair	Nltk	Title	I/O
Trigram	Vader		Case	
5-gram			Stopword	

N-Grams:

Unigram, Trigram, 5-Gram

Sentiment Analysis:

Two pretrained models:

- Flair
  - Pretrained on builtin IMDB data
  - Features: {Sentiment: positive or negative, Score: 0-1.0}
- Vader
  - nltk implementation
  - Features: {Neg: 0-1.0 , Pos: 0 - 1.0, Composite:0-1.0}

Scope:

- Sentence Level: feature corresponding to the sentiment models prediction on the entire sentence the token comes from
- Token Level: n features corresponding to the sentiment models prediction on each of the n tokens in the n-gram

Part of Speech Tags

nltk - Penn Treebank

Title Information:

Binary: 1 if token is also in title string

Case Information:

Binary: 1 if token was capitalized prior to lowercase normalization

Stop Word:

Binary: 1 if token was a stopword

I/O Encoding:

- Class Encoding: Propagpanda (p) or Other (o)

6 Testing Phases

Testing occurred in three phases. Each phase had 3 feature sets, one for each n-gram dimension. Each of these feature sets was trained and tested with 5 optimization algorithms for a total of 45 individual tests. Phase 1 tested each of the n-grams without sentiment information. Phase 2 tested the addition of sentiment information. Phase 3 tested only 5-grams with the addition of sentiment information to a subset of tokens in the 5-gram. Preliminary testing was conducted on random subsets of all of the features to inspect the impact they had on prediction. Both title information and sentence level sentiment information always had a negative impact on prediction so they were discarded. There was a negligible difference in performance between the Vader and Flair sentiment models. Vader became the default only because it was substantially faster at runtime.

Test 1 Feature Set: POS, Case Information, Token, Stopword Information

Test 2 Feature Set: Test 1 Set, Token Level Sentiment Information

Test 3 Feature Set: Test 1 Set, Varying levels of Token Level Sentiment Information (only 5-grams)

Training and testing data was split 80/20. The same splits were maintained for each algorithm and test phase. L-BFGS and l2sgd were both trained at a thousand iterations. Arow, ap, and pa were all trained at 100 iterations. All of the algorithms had a convergence stopping criteria of 1e-5. For each training/test set, sentences were reduced to lists of tokens which were then tagged with appropriate features. The objective for each model then is to decide the best sequence of tags (propaganda or other) for the given sequence of tagged tokens.

7 Results

In order to examine the results, graphs for each test phase and feature combination are provided below. The results refer to the f1, recall, and precision scores for only predictions on propaganda labels and not on accuracy scores related to predictions of the 'other' label. Propaganda is significantly underrepresented in the PTC corpus so accuracy scores related to 'other' are artificially inflated (>= 0.9). This under-representation problem is one of the main

pitfalls and discussed at length in the section following the graphs and results data.

As far as results, ranking was done according to F-1 score. Arow optimization regularly outperformed all other optimization algorithms for each test phase. It was able to obtain a .28 F1-Score for all configurations in each test set except for the Unigram models in test phase 1 and 2. Overall it appears most of the optimization algorithms favored higher precision over recall, and that arow had a slightly stronger bias toward recall which lead to its superior performance. Generally the models improved performance as the n-gram range was increased. All models except for l2sgd had their highest F1-Scores with 5-grams. L2sgd regularly had a Unigram score that was higher than its Trigram score.

Sentiment information overall had mixed effects. The arow algorithm maintained similar scores regardless of the addition of sentiment information, whereas other algorithms were plus or minus 1-2% with the addition. Overall it appears the sentiment model contributed little to performance.

Precision	Recall	F1-Score	Optimization	N-Gram	Test Phase
0.42	0.11	0.17	lbfgs	5	test1
0.45	0.10	0.17	lbfgs	Trigram	test1
0.40	0.08	0.13	lbfgs	Unigram	test1
0.52	0.09	0.16	ap	5	test1
0.54	0.07	0.13	ap	Trigram	test1
0.59	0.04	0.08	ap	Unigram	test1
0.57	0.05	0.10	l2sgd	5	test1
0.61	0.02	0.04	l2sgd	Trigram	test1
0.46	0.05	0.10	l2sgd	Unigram	test1
0.52	0.12	0.19	pa	5	test1
0.52	0.10	0.16	pa	Trigram	test1
0.52	0.06	0.11	pa	Unigran	test1
0.28	**0.28**	**0.28**	arow	5	test1
0.29	0.27	**0.28**	arow	Trigram	test1
0.28	0.21	0.24	arow	Unigram	test1

Precision	Recall	F1-Score	Optimization	N-Gram	Test Phase
0.40	0.11	0.18	lbfgs	5	test2
0.44	0.11	0.17	lbfgs	Trigram	test2
0.42	0.09	0.15	lbfgs	Unigram	test2
0.54	0.08	0.14	ap	5	test2
0.55	0.06	0.11	ap	Trigam	test2
0.60	0.04	0.07	ap	Unigram	test2
0.54	0.07	0.12	l2sgd	5	test2
0.57	0.01	0.02	l2sgd	Trigram	test2
0.54	0.04	0.08	l2sgd	Unigram	test2
0.54	0.10	0.17	pa	5	test2
0.54	0.09	0.15	pa	Trigram	test2
0.52	0.06	0.11	pa	Unigram	test2
0.29	**0.28**	**0.28**	arow	5	test2
0.29	0.26	**0.28**	arow	Trigram	test2
0.29	0.19	0.23	arow	Unigram	test2

Precision	Recall	F1-Score	Optimization	N-Gram	Test Phase
0.41	0.11	0.18	lbfgs	n+2, n-2	test3
0.41	0.12	0.19	lbfgs	n+1, n-1	test3
0.42	0.12	0.19	lbfgs	n	test3
0.52	0.08	0.14	ap	n+2, n-2	test3
0.53	0.08	0.14	ap	n+1, n-1	test3
0.53	0.09	0.16	ap	n	test3
0.44	0.11	0.17	l2sgd	n+2, n-2	test3
0.41	0.18	0.25	l2sgd	n+1, n-1	test3
0.43	0.09	0.15	l2sgd	n	test3
0.52	0.11	0.18	pa	n+2, n-2	test3
0.53	0.11	0.19	pa	n+1, n-1	test3
0.52	0.12	0.19	pa	n	test3
0.29	**0.28**	**0.28**	arow	n+2, n-2	test3
0.29	**0.28**	**0.28**	arow	n+1, n-1	test3
0.29	0.27	**0.28**	arow	n	test3

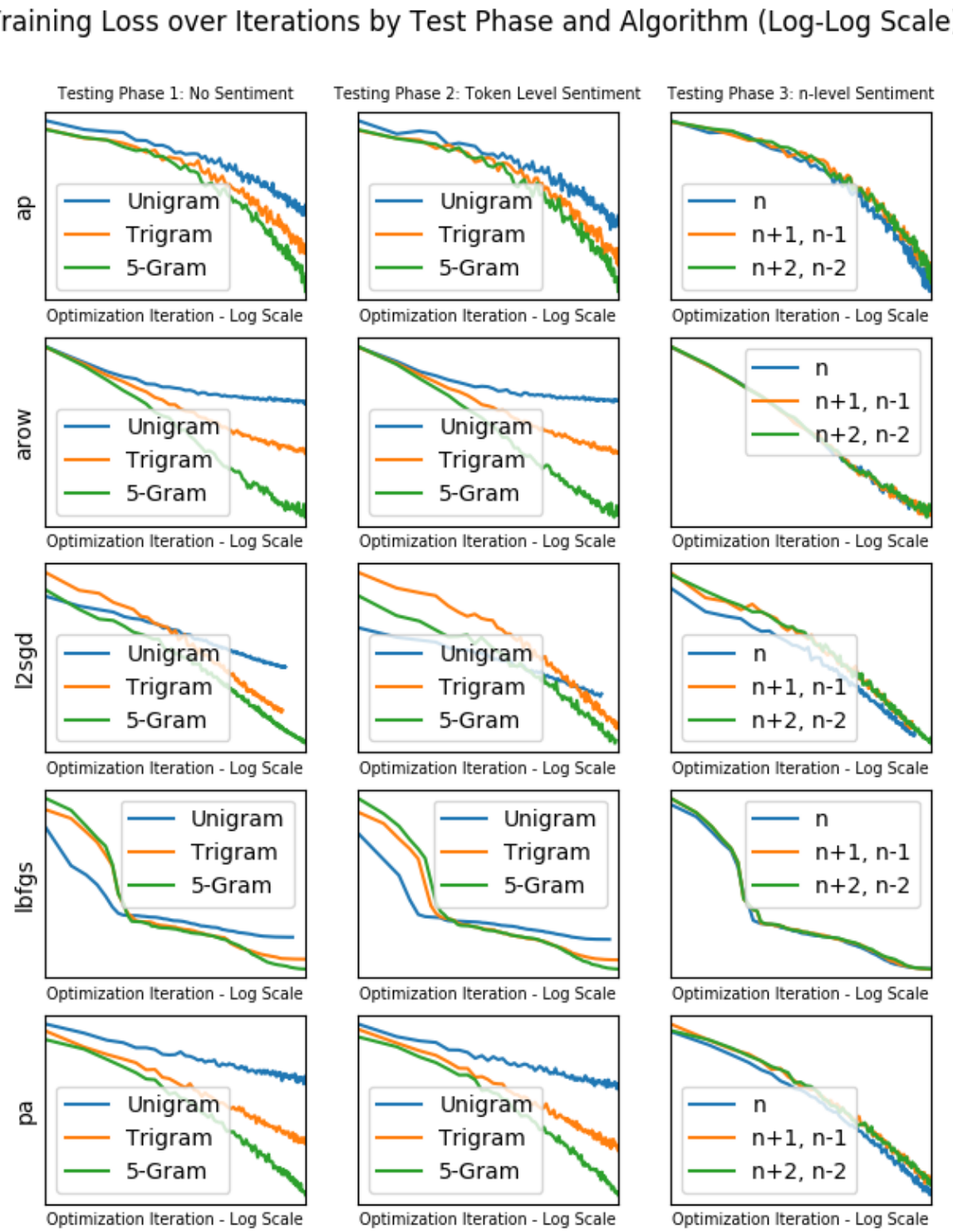
Training Loss over Iterations by Test Phase and Algorithm (Log-Log Scale)
Testing Phase 1: No Sentiment
Testing Phase 2: Token Level Sentiment
Testing Phase 3: n-level Sentiment
ap
arow
l2sgd
lbfgs
pa
Unigram
Trigram
5-Gram
n
n+1, n-1
n+2, n-2
Optimization Iteration - Log Scale

Figure 1

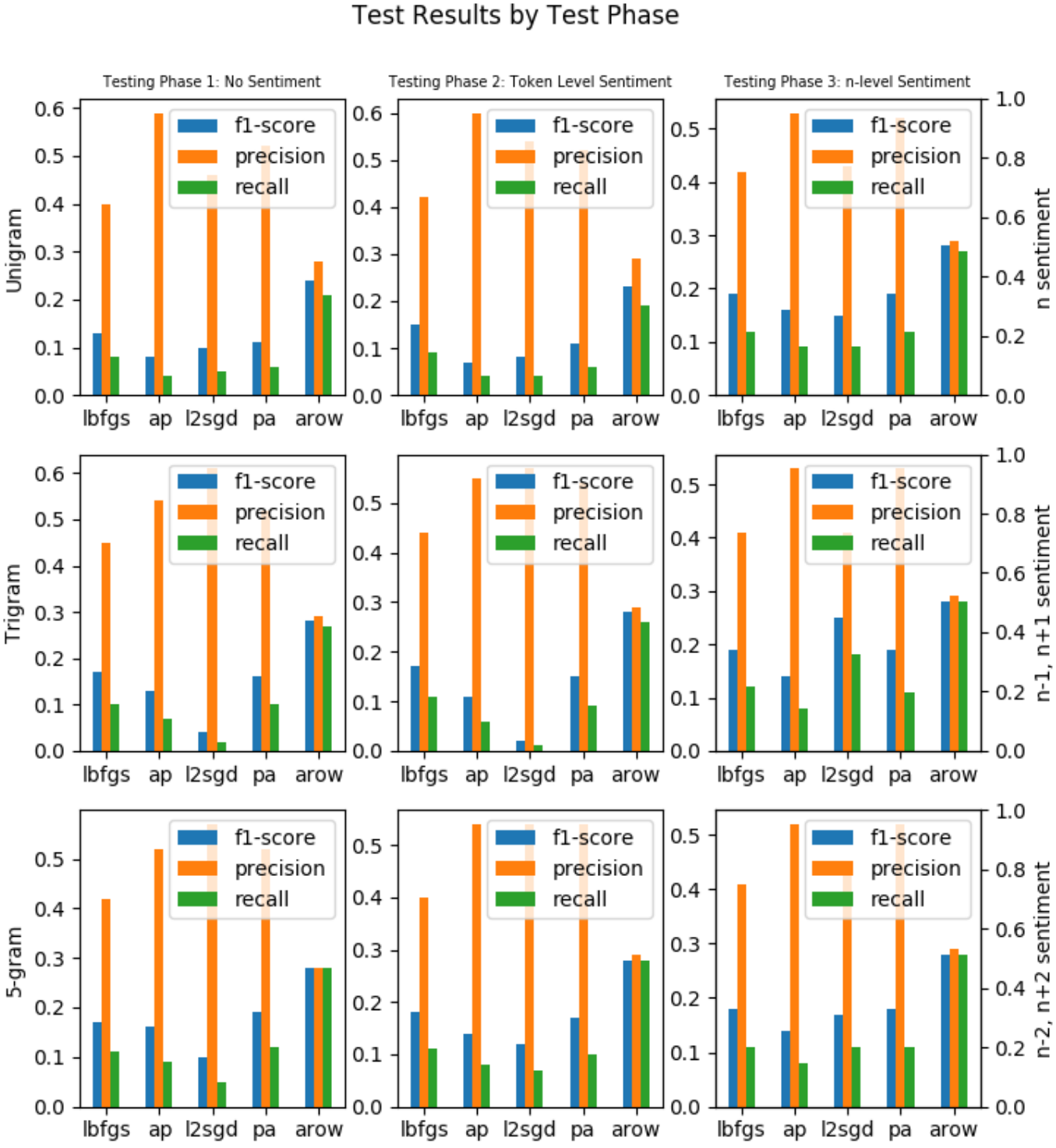
Test Results by Test Phase
Testing Phase 1: No Sentiment
Testing Phase 2: Token Level Sentiment
Testing Phase 3: n-level Sentiment
Unigram
Trigram
5-gram
n sentiment
n-1, n+1 sentiment
n-2, n+2 sentiment
f1-score
precision
recall
lbfgs
ap
l2sgd
pa
arow

Figure 2

8 Future

After the previous tests I now have an established baseline from which to work from. I have identified three main areas in which the model could be improved.

Model Architecture:
There are a host of different model architectures that could possibly improve performance. Recurrent Neural Networks and their combination with CRFs seems to be the most logical step going forward. Incorporating token embeddings into these models along with some of the manually constructed features from the prior tests seems fairly feasible. I initially incorporated token embeddings into the CRF model but backed off because I had no way of measuring if the perfomance would be only due to the high-dimensional nature of the data. Now that I have a substantial base model I will be able to fairly judge improvements.

Sampling & Handling Under-representation:
One of the biggest hurdles facing Propaganda Technique Identification is the unbalanced nature of the data set. The propaganda class is severely underrepresented in the corpus which makes training difficult. The models can be relatively accurate by essentially guessing the most frequent class in the data set. For the propaganda identification task this means the models have a bias against classifying a token as propaganda. I have begun exploring two methods for addressing this. The first method involves using SMOTE(Chawla 2002) on the sequence data. The second method is to create artificial spans of propaganda by replacing spans from the original data set with synonyms and adding them to the data set. While both of these ideas are appealing they will be the last method I approach as I feel they leverage the statistical nature of the data rather than actual aspects of propaganda.

Data Specific Features and Sentiment Models:
There are two propaganda-specific features that I think could be beneficial for prediction. Some kind of token collocation information could be added. During preliminary tests title information was found to be detrimental to prediction, however after reviewing some of the propaganda technique descriptions, it becomes apparent that repetition of individual tokens may be a marker for certain techniques. Finding a way to leverage this might lead to better prediction. Second, additional analysis needs to be performed on what types of propaganda the models are correctly identifying. In task 2 of this competition each of the spans from task 1 are identified with a respective technique. This information could help identify the specific propaganda techniques the models are missing. Additional semantic and syntactic features could then be tailored

for these specific techniques. In addition to this, it is now clear that the sentiment model is not having a significant impact on performance regardless of the architecture or feature combination. This leads me to believe that both the Vader and Flair models either need to be fine-tuned for the PTC corpus or discarded. It was initially believed that the sentiment analysis would provide some kind of reliable metric for emotional intensity that the models could leverage but this has not been the case.

References

1. (Andrew 07) Galen Andrew and Jianfeng Gao. "Scalable training of L1-regularized log-linear models". Proceedings of the 24th International Conference on Machine Learning (ICML 2007). 33-40. 2007. (Chawla 2002) Nitesh V. Chawla and Kevin W. Bowyer and and Lawrence O. Hall and W. Philip Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique", Journal of Artificial Intelligence Research 16 (2002), 321 – 357. Submitted 09/01; published 06/02.
2. (Collins 02) Michael Collins. "Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms". Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2002). 1-8. 2002.
3. (Crammer 06) Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. "Online Passive-Aggressive Algorithms". Journal of Machine Learning Research. 7. Mar. 551-585. 2006.
4. (Flair 2018) Akbik, Alan and Blythe, Duncan and Vollgraf, Roland "Contextual String Embeddings for Sequence Labeling" COLING 2018, 27th International Conference on Computational Linguistics, 2018
5. (Jurafsky 2009) Jurafsky, Daniel and Martin, James H., Speech and Language Processing (2nd Edition) Prentice-Hall, Inc.USA 2009
6. (Lafferty, 2001) John Lafferty, Andrew McCallum, and Fernando Pereira. "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data". Proceedings of the 18th International Conference on Machine Learning. 282-289. 2001.
7. (Martin 2019) G. Da San Martino, S. Yu, A. Barrón-Cedeño, R. Petrov and P. Nakov, "Fine-Grained Analysis of Propaganda in News Articles", to appear at Conference on Empirical Methods in Natural Language Processing (EMNLP 2019), Hong Kong, China, November 3-7, 2019.

8. (Martino 2019) Giovanni Da San Martino, Alberto Barrón-Cedeño, Preslav Nakov, "Evaluation of Propaganda Detection Tasks Shared Task at SemEval 2020 Task 11: "Detection Of Propaganda Techniques In News Articles"". email: gmartino@hbku.edu.qa
9. (Shalev-Shwartz 07) Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. "Pegasos: Primal Estimated sub-GrAdient SOlver for SVM". Proceedings of the 24th International Conference on Machine Learning (ICML 2007). 807-814. 2007.