**Literature Notes**

**Paper 1 – “Toxicity Detection: Does Context Really Matter?”**

* Pavlopoulos J., Sorensen J., Dixon L., Thain N., and Androutsopoulos I. (2020). “Toxicity Detection: Does Context Really Matter?”, in *Proc. of 58th Annual Meeting of Association for Computational Linguistics,* pp. 4296-4305.
* <https://www.aclweb.org/anthology/2020.acl-main.396.pdf>
* Most datasets ignore context of posts, judging comments independently, without preceding comments – so systems ignore context when trained on dataset
* No other statistics on how often context affects perceived toxicity
* Context = parent comment + thread title
* 250 comments annotated with/without context by 2 groups – average toxicity scores of annotators taken - used to make binary decision of toxic or not
* Perceived toxicity of significant subset of posts changes when context is/isn’t provided
* Context amplified toxicity of 3.6% of comments, mitigated toxicity of 1.6%
* 20k comments – ½ with context, ½ without used to train classifiers – tested on separate set of comments with context (unbalanced – toxic comments rare)
* No evidence context improves performance of toxicity classifiers (range of context-aware classifiers and mechanisms used) – Assumed related to small number of context-sensitive comments
* Toxicity and subtypes strongly related - toxicity detection systems also effective against subtypes (e.g. hateful language)
* State of the art language modelling – use LDA to encode preceding sentences + pass history to RNN language model – alternative solution to LSTMs to solve vanishing gradients (Mikolov and Zweig (2012), Blei et al. (2003))
* Adding more preceding comments led to annotators ignoring context completely
* Statistically significant increase in proportion of comments that are toxic when context given (toxicity ratio increases by 2% with context – aggregated result) (on 250 dataset)

(0.9% increase on 20k dataset)

* Need to find larger annotated datasets to estimate frequency of context-sensitive posts
* Context-insensitive classifiers – bidirectional LSTM - added feed-forward neural network to concatenated last states – relatively simple classifier, BERT (bidirectional encoder representations from transformations)– higher complexity – fine-tuned on training subset w/ FFNN on top (Delvin et al., 2019), BERT-CCTK – model same but tuned on sample of Civil Comments dataset, PERSPECTIVE – CNN-based model trained on millions of comments from online publishers (publicly available – can’t be modified for context)
* Context sensitive classifiers – CA-BILSTM-BILSTM – context aware extension of BILSTM - added 2nd BILSTM for parent comment – vector representations passed to FFNN, CA-BILSTM-BERT – use BILSTM to encode parent – extension of BERT classifier – results passed to FFNN, CA-SEP-BERT – context-aware BERT – concatenates parent + target (no separate encoder for parent), CA-CONC-BERT-CCTK, CA-CONC-PERSPECTIVE – concatenate parent and target at test time (naïve)
* PERSPECTIVE + BERT-CCTK and CA counterparts performed best – trained on largest toxicity datasets
* Future – larger annotated datasets with context, look at specific topic/tone/group, add more than just title + parent comment for context
* Toxicity rating – (Perspective guidelines) – Very Toxic, Toxic, Hard to Say, Not Toxic

**Paper 2 – “Nuanced metrics for measuring unintended bias with real data for text classification”**

* Borkan D., Dixon L., Sorensen J., Thain N., and Vasserman L. (2019). “Nuanced metrics for measuring unintended bias with real data for text classification”, in *Companion Proceedings of the 2019 World Wide Web Conference*,Association for Computing Machinery, pp. 491–500.
* <https://storage.googleapis.com/pub-tools-public-publication-data/pdf/66073ca7ac60ee38e93fc1d173a09cab65f2fef3.pdf>
* Unintended bias leads to systematic differences in performance for different demographic groups – seen when model performance varies across set of groups (skewed classifier scores) (assumes reliable labelling of groups)
* Toxicity = “a rude, disrespectful, or unreasonable comment that is likely to make you leave a discussion”
* Toxicity models shown to capture + reproduce societal biases – mis-associate names for frequently attacked groups with toxicity – due to demographic composition of user pool/biases of those labelling/selection of items to label
* Synthetic test set w/ reliable labels + large human-annotated test set w/ high rating redundancy (Civil Comments Toxicity Kaggle)
* Threshold dependant metrics can obscure unintended bias, threshold agnostic metrics capture behaviour of underlying model – single metrics obfuscate essential info so use suite of 5 (each captures different aspect of model performance) – new metrics robust to class imbalances
* Uses Perspective API models – TOXICITY@1 (significant unintended bias around ‘gay’, ‘transgender’) and TOXICITY@6 (trained using bias mitigation technique for short comments – reduced not eliminated unintended bias) – imbalances in toxicity in training data for certain identity words are major source of bias (most prevalent in short comments) – additional training data added to even out toxicity prevalence
* Equality Gap evaluates model only at one specific threshold so falls short (diff between TPR of subgroup and that of background at specific threshold)
* AUC-based metrics – measure probability randomly chosen negative example will score lower than positive (correctly ordered) – threshold agnostic – new metrics measure variations in distributions that cause mis-orderings – can identify if false positives/negatives likely when threshold selected – robust to data imbalances in positive and negative examples in test set
* Compare subgroup to rest of (background) data – e.g. score shift for subgroup
* Subgroup AUC = AUC (examples in subgroup) – represents model understanding and separability within subgroup
* Background Positive Subgroup Negative (BPSN) AUC = AUC (positive examples in rest of data + negative examples in subgroup) – reduced when negative subgroup scores > positive examples (would appear as false positive within subgroup for many thresholds) (score shifts)
* Background Negative Subgroup Positive (BNSP) AUC = AUC (negative examples in rest of data + positive examples in subgroup) – reduced when positive subgroup scores < negative examples (would appear as false negatives within subgroup for many thresholds)
* Average Equality Gap (AEG) [-0.5,0.5] (optimal when 0 – Equality of Opportunity holds for every threshold) – generalisation of Equality Gap – threshold agnostic – plots TPR of subgroup and background against each other for every possible threshold t – AEG captures average bias against all thresholds for classifier
* Positive AEG – area between curve (x(t),y(t)) and line y = x (TPRs of subgroup and background) (also ½ - prob positive example from background higher than positive example from subgroup), Negative AEG – uses TNR instead
* Mann-Whitney U – rewrite PAEG – efficient closed form for computing PAEG, (same for NAEG), all definitions of AEG equivalent
* Biases: small score shift – only AEGs notice bias, large score shift – ideal threshold for background means false positives in subgroup so low BPSN AUC and high AEG metrics, score shift + size skew (more positives in subgroup) – noticed by BPSN, BNSP AUCs and AEGs, left score shift – negative AEGs, low BNSP AUC (more false negatives in subgroup), low subgroup separability (classifier underperforms on subgroup relative to background) – noticed by Subgroup AUC metric and AEG metrics, wide subgroup score range (no overlap) – higher variance of scores for subgroup - no bias noticed by metrics (could be problematic depending on use case), wide subgroup score range (w/ overlap) – AUC metrics notice bias
* Synthetic test set - bias towards toxicity for certain groups – high toxicity for non-toxic examples with words like ‘homosexual’, ‘gay’, score distributions vary widely across groups
* Human labelled dataset – imbalance in toxicity between identities – e.g. 8% of all comments toxic, 28% of comments about homosexuals toxic; results vary between short and long comments (look at comment length); more unintended bias than synthetic data; bias skews towards toxicity (may be due to societal perceptions of online conversation – identities with most bias are most frequently attacked)
* Future work: choosing optimal threshold, evaluating CCTK vs sub-string matching of identities, systematic definition of synthetic distributions for evaluating metrics for unintended bias, full taxonomy of different possible biases + systematic approach for metrics in diagnosis

**Paper 3 – “Classifying Constructive Comments”**

* Kolhatkar V., Thain N., Sorensen J., Dixon L., and Taboada M., (2020). “Classifying Constructive Comments”. *arXiv preprint arXiv:2004.05476*.
* <https://arxiv.org/pdf/2004.05476.pdf>
* Constructive comments – high-quality comments that make a contribution to the conversation, opinion with justification/evidence
* Promote constructive comments (proactive intervention) rather than filtering out undesirable comments (reactive interventions) – positive contagion effect (more constructive comments leads to more constructive comments)
* Respect instead of like button – engage with different opinions (Stroud (2011))
* Classifying comments – non-constructive (insulting) “Another load of tosh…”, non-constructive (positive) “Another wonderful read!...”, opinion (no justification), constructive (toxic) – adds to conversation but toxic, constructive – reasoned opinion, supported by personal experience
* No context/metadata – evaluate comment on merit alone – can also look at degree of connection between comment + article (relevance)
* Naïve models have length as most important feature (limited practical value) - NYT picks 127.2 words per comments (81.7 for non-picks) – CNNs and transformer-based models robust
* Feature-based classifiers – (sklearn w/ stochastic gradient descent) SVMs/logistic regression – features: char + word n-grams, average word length, comment length, linguistic features, argumentation, named entities, readability, content quality, aggressiveness, toxicity + toxicity scores from Perspective (content quality, aggressiveness + toxicity features) + no. spelling mistakes, capitalised words, punctuation tokens
* Word embeddings popular – averaging pre-trained word embeddings/contextual using paragraph2vec; deep learning – RNNs/CNNs
* Sentiment analysis/polarity of words not useful for constructive comments
* Using C3 dataset – annotation scheme (no. constructive characteristics + absence of non-constructive characteristics)– if contributor agreed with comment, constructive: provides solution, targets specific points, evidence, personal story, encourages dialogue (most important predictor), non-constructive: not relevant (important predictor), no respect for views of others, unsubstantial (important predictor), sarcastic, provocative (used logistic regression to determine usefulness of criteria – F1 score 0.87), 80% train, 20% test
* Inter-annotator agreement – 66.57% instances had unanimous agreement, 10% serious disagreement; average chance-corrected inter-annotator agreement for binary classification 0.71 (better than other datasets including toxicity); expert agreed with crowd 87% of time – disagreements crowd labelled constructive, expert did not (not relevant enough/no dialogue)
* Moderate correlation between constructiveness + agreement with view (Pearson = 0.56) – looked into – constructiveness qualitatively different from agreement
* Constructiveness and toxicity different features – orthogonal (independent)
* Deep learning classifiers (generally best performing) – BILSTMs, CNNs (w/ GloVe embeddings), BERT – performance drops (compared to feature-based) when test + train in different domains – CNN less dependant on length due to max-pooling layer (doesn’t overfit) – need length insensitive models to overcome length imbalance in data – flexible so benefit from being trained on whole dataset – inbuilt resistance to overfitting; CNN had 1 embedding layer (pretrained GloVe dim 300 for input word tokens), 1 convolutional (128 filters: size 3,4,5) and pooling (max-pooling across sentence), 1 fully connected (produces 1 value per class); BILSTM had 1 embedding layer (w/ GloVe), 1 recurrent (biLSTM w/ cells size 128), 1 fully connected; BERT on top of variant of pretrained BERTBASE, output fed into 3-layer fully connected NN (layers 256,128,64); dropout 0.5, Adam optimizer – learning rate 0.001
* Compared C3 to SOCC-a, NYT, YNACC\* - C3 good training for SOCC-a test set (SVM)
* Length best predictor of constructiveness for feature-based classifier (skewed distribution in constructive/non-constructive comments) (less important when using multiple contexts/domains (different test + train)) – 0.65 (high) correlation between length and constructiveness – not generalisable for constructiveness + vulnerable (start writing long low-quality comments) – over-dependence on length (FP higher length than FN), text quality and all features next most important – toxicity/aggressiveness not good measure for constructiveness; text quality and lexical features important in domain transfer

**Paper 4 – “Deceiving Google’s Perspective API Built for Detecting Toxic Comments”**

* Hosseini H., Kannan S., Zhang B., and Poovendran R. (2017). “Deceiving Google’s perspective api built for detecting toxic comments”, *arXiv preprint, arXiv 1702.08138.*
* <https://arxiv.org/pdf/1702.08138.pdf>
* *Note: running same examples more recently shows that Perspective API is now more robust to below attacks*
* Adversarial examples – change algorithm by subtly perturbing input – effective even when adversary only has black-box access to target model
* Modified texts that contain same highly abusive language but receive lower toxicity score; misspell words, add punctuation between letters
* Perspective API – millions of comments from different publishers, asked panels of 10 people to rate comments on scale from “very toxic” to “very healthy” contribution; real-time toxicity scores
* Adversary can query model and get toxicity score; same modification reduces toxicity score for different phrase; can make dictionary of adversarial perturbations
* False alarm – adding not to toxic phrases doesn’t reduce toxicity (rightly so?)
* Somewhat robust to phrases containing randomly modified toxic words
* Vulnerable to poisoning attack – allows users to provide feedback on toxicity scores; modifies training data so model assigns low toxicity scores to certain phrases
* Solutions – adversarial training – on correctly labelled adversarial examples, spell checking (may increase false alarm), blocking suspicious users temporarily (so can’t try different error patterns on system)

**Paper 5 – “WikiDetox Visualisation”**

* Qu I., Thain N. and Hua Y., “WikiDetox Visualization”.
* <https://wikiworkshop.org/2019/papers/Wiki_Workshop_2019_paper_17.pdf>
* Used Wikipedia datasets to train Perspective API
* Toxicity subtypes: flirtation, threat, identity attack, insult, sexually explicit, obscene, and severe toxicity
* Metadata: page title, timestamp, user id – unclear how conversation unfolded (edits to page) so conversation reconstructed, action types: creation, addition, modification, restoration, deletion, creates new reconstructed attributes
* Toxic comments often off-topic
* Categorized pages using Google Cloud Natural Language API <https://cloud.google.com/natural-language/>; got direct subcategories, 3 most relevant categories returned (confidence level > 0.5)
* User pages always more toxic than article pages; theory toxic actions on article carries over to user pages
* Most toxic categories (2017): People & Society, Arts & Entertainment, News and Law and Government; sub-categories: Music & Audio, People & Society, Movies, Politics; undeleted comments only (May 2018): Religion + Belief, Biological Sciences, Movies, Politics

**Paper 6 – “Ex Machina: Personal Attacks Seen at Scale”**

* Wulczyn E., Thain N., and Dixon L. (2017). “Ex machina: Personal attacks seen at scale”, in *ICWWW*, pp. 1391–1399.
* <https://arxiv.org/pdf/1610.08914.pdf?_gclid=5aec59ba53a138.82841565-5aec59ba53a189.59055081&_utm_source=xakep&_utm_campaign=mention114889&_utm_medium=inline&_utm_content=lnk530117377130>
* English Wikipedia dataset; labelled 100k according to whether personal attack or not – trained classifier using test set to get 63M comments machine-labelled from classifier (as good as aggregate of 3 crowd workers (AUC and Spearman))
* Character n-grams (1-5) result in flexible and performant classifier (higher robustness to spelling variations (common, especially in expletives))
* Empirical distribution of human ratings produces better classifier than majority vote; classifier threshold balanced precision and recall
* Main approaches in sentiment analysis/spam detection; classifiers - SVM on sentiment and context features, separate classifiers for separate identity groups, random forests + logistic regression to predict banned users
* Attributes: personal attack? if yes, whether attack has target/quotes previous attack
* Crowd-sourcing – use question with highest inter-annotator agreement on 1000 comments; used annotators with > 70% accuracy on test set (excluded worst 2% contributors); each comment labelled by at least 10 annotators; 0.45 inter-annotator agreement (others now better); mostly not unanimous judgements
* Roughly 1% of comments on Wikipedia Talk pages are personal attacks; enhanced dataset by sampling comments made by blocked users (17% personal attacks)
* Only using comment features; no context
* Classifiers – logistic regression, MLPs, (LSTMs in future); bag-of-words representations; character/word n-grams (more powerful than linguistic/syntactic features/lexicons/word embeddings); always final softmax layer and cross-entropy as loss function
* Best performing was MLP with character n-grams using ED labels
* 3:1:1 train, development, test; 15 iterations of random search over grid of relevant hyper-parameters; metrics – AUC and Spearman for ED labels
* Annotations – binary classification so one-hot encoding determined by majority of annotators; annotations for comment form approximate empirical distribution (ED) – label [0.3,0.7] instead of [0,1] – trained each architecture using OH and ED labels
* Anonymity can lead to heightened aggression and inappropriate behaviour – attack prevalence among comments by anonymous users 6 times as high as registered users but only 9.6% of comments in dataset from anonymous users; most comments made by very active users but activity level doesn’t change % of comments that are attacks; 80% of attacks from 9000 users with fewest attacking comments, 9% of attacks by 34 most toxic users; comments surrounding an attack 22 times more likely to be an attack than for a non-attacking comment – attacks cluster in time – early intervention by moderators could help; <1/5 trigger blocking
* longitudinal data – same sample, different points in time
* Cleaning Wikipedia data – using revision history of comments (not complete snapshot)
* 20-50% comments administrative – (most) filtered out from sample

**Paper 7 – “Benchmarking Aggression Identification in Social Media”**

* Kumar R., Ojha A. K., Malmasi S., and Zampieri M. (2018). “Benchmarking aggression identification in social media”, in *TRAC*, Santa Fe, USA.
* <https://www.aclweb.org/anthology/W18-4401.pdf>
* Shared ML task – many teams given data + tried to find best model
* Classes: overtly aggressive, covertly aggressive, non-aggressive, (identities: gender, religion, caste, country of origin, race (not mutually exclusive – multi-label classification problem) (can be classified with surface-level linguistic features))
* Dataset: 15,000 annotated Facebook posts and comments in Hindi/English; some English comments contain code-mixed Hindi-English data + other languages;
* Test set: more FB posts and some tweets (tested on different domain)
* Most research focuses on a specific toxicity subtype – duplication of research, lack of focus and reusability of datasets
* (related work) Categories: target (individual/group), nature of language (explicit/implicit); individuals and groups often targeted simultaneously
* Classifiers: 1) (**best)** LSTM (RNN and CNN as features), translation for data augmentation, pre-processing – correcting spellings, translating emojis, sentiment score; 2) Passive-Aggressive and SVM classifier combination, character based n-gram (1-5), TF-IDF feature representation; 3) LSTM w/pretrained fasttext vector for embeddings and a CNN; 4) biLSTM w/ GloVe embeddings; 4) voting-based ensemble method w/ CNN (4 layers), LSTM, biLSTM; 5) translation for data augmentation, ensemble of TF-IDF approaches, character n-grams (2-6), word n-grams (1-2) w/ bidirectional RNN w/ fasttext embeddings; 6) stacked ensemble (SVM on top of SVMs) trained on 1-6 character n-grams and word unigrams, also plain SVM w/ character and word bag-of-n-grams (overlapping character + word n-gram features weighted with sublinear TF-IDF, tuned using 5-fold CV, case normalisation, SVM regularisation param C; 7) novel deep-learning based on multi-task learning – evaluated with 3 NN models, multiple convolution structure w/ trainable embedding layer; 8) soft voting of RNN (3 pre-processed feature sets – GloVe, sentiwordnet, TF-IDF + Vader Sentiment analysis) and SVM (TF-IDF of post stemmed terms, excluding stop words, 3-5 character n-grams); 9) logistic regression w/ pre-processing (removing non-ascii characters, stop words, replacing new line, n’t with not), 1-3 word n-grams, 2-6 character n-grams; 10) logistic regression w/ pre-processing (correct spelling, replace URLS, numbers, email addresses), word unigrams, char (4-5), TF-IDF, w/ Google new pre-trained word embedding; 11) random forest, SVMs, 300 semantic features, sentiment scores; 12) dense neural networks; 13) SVM, deep NNs; 14) combination of doc2vec and logistic regression, combination of CNN, LSTM; 15) LSTM w/ attention and simple embeddings (word to index); 16) single channel CNN e/ Bayesian optimisation for tuning; 17) winner-takes-all autoencoder, input dim log-normalised, sentiwordnet-score weighted word-count vector, binary cross-entropy loss function; 18) open vocab approach and ensemble model of Naïve Bayes w/ CountVectorizer for pre-processing and RNN w/ 1 embedding layer and 2 LSTM layers w/ soft voting; 19) Multinomial naïve bayes, unigrams, bigrams, trigrams, chi^2 test for features + features from LIWC2015; 20) combines NN and new word representation model (trains back propagation n NN); 21) pooled recurrent unit architecture e/ pre-trained word embeddings, vectors aligned w/ pre-computed SVD matrices – pulls representations from different languages into single space; 22) random forests, augmented with CCTK; 23) ensemble of CCN 2D w/ MAXPOOL and SVM classifiers, passed through 3 dense layers to predict output, softmax outer layer; 24) random forest e/ surface-level features (no. lines, uppercase + lowercase letters, digits, named entities, Unicode characters…); 24) combination of 12 distance measures, kNN and canonical genetic algorithm; 25) unsupervised, based on multilingual lexicon of aggressive words, BabelNet; 26) biLSTM
* fasttext – library for efficient text classification + representation learning

**Paper 8 – “ConvAI at SemEval-2019 Task 6: Offensive Language Identification and Categorization with Perspective and BERT”**

* Pavlopoulos J., Thain N., Dixon L., and Androutsopoulos I. (June 2019). “Convai at semeval-2019 task 6: Offensive language identification and categorization with perspective and bert”, in *Proceedings of the 13th International Workshop on Semantic Evaluation*, pp. 571-576.
* <https://www.aclweb.org/anthology/S19-2102.pdf>
* Perspective better than BERT at detecting toxicity, BERT better at categorising offensive type (both strong baseline classifiers)
* Perspective API – CNN trained on millions of user comments from different online publishers – no extra training/fine tuning – can be directly applied; used pre-trained models from API – toxicity model – CNN based on GloVe word embeddings – robust, somewhat adaptable to different datasets; other models for categorisation – identifies: toxicity, insult, threat, profanity, identity attack , attack on commenter (averaged some scores to calculated offensiveness)
* BERT (bidirectional encoder representations from transformers) – pre-trained model – state- of-the-art performance in NLP tasks; limited fine-tuning on task-specific training data; deep bidirectional network built using transformers – pre-trained to detect masked word from context and next sentence; using BERT Base – 12 transformer layers, 768 hidden states – can add task-specific layer for fine-tuning (dropout, linear transformation, softmax)
* OLID dataset – subtasks: detecting offensive language, categorising offensive language as targeting entity or not, identify if target is individual or group
* Offensive tweets targeting group lengthier – 28 tokens on average
* Both classifier affected by class imbalance

**Paper 9 – “The Risk of Racial Bias in Hate Speech Detection”**

* Sap M., Card D., Gabriel S., Choi Y., and Smith N.A. (July 2019). “The risk of racial bias in hate speech detection”, in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 1668-1678.
* <https://homes.cs.washington.edu/~skgabrie/sap2019risk.pdf>
* Automated removal of content risks further supressing marginalised voices
* “Toxic” depends on social context (speaker’s identity/dialect) – terms used to disparage communities have been reclaimed by them (but offensive for outsiders to use)
* African American English dialect (AAE) – labelled by Perspective as much more toxic than American English equivalents (models propagate bias); Perspective API shows racial bias
* Racial bias in widely used Twitter corpora; strong associations between AAE markers and toxicity annotations (especially “offensive” and “abusive” labels); differences between FPs for AAE/white-aligned groups; strong tendency to label white tweets as ‘none’; violates equality of opportunity criterion
* Mitigate annotator bias – dialect and race priming – design tasks that explicitly highlight inferred dialect of tweet/likely racial background of author (annotators less likely to label as offensive with this info)
* Infer dialect using lexical detector of words associated with AAE/white-aligned English – topic model (Blodgett et al. (2016)) trained on 60M tweets and US census race/ethnicity data for topics
* Davidson et al. (2017) and FDCL18 as training datasets
* Apply models trained on biased datasets to 2 reference twitter corpora: Demographic16 and UserLevelRace18; classifier – basic neural attention architecture; initialised with GloVe vectors to minimise cross-entropy of annotated class conditional on text; biLSTM with attention + projection layer
* Annotators given likely racial background – asked if tweet could be offensive to them/anyone – took gender, age, race, political leaning of annotators (skewed demographics – 75% white); 5 annotations for each of 1351 tweets; 76% pairwise agreement on offensive tweets (k = 0.48)
* Dialect and race priming significantly reduces AAE tweet’s likelihood of being labelled offensive; motivations to not seem prejudiced could buffer stereotype use – could influence annotator response
* Most highly weighted unigrams for predicting ‘hateful’ speech are variations of word strongly associated with AAE; correlations robust even when removing tweets with these terms

**Paper 10 – “Toxic Speech Detection”**

* Koratana A. and Hu K. (2019). “Toxic Speech Detection”.
* <http://web.stanford.edu/class/cs224n/reports/custom/15744362.pdf>
* Code: <https://goo.gl/GgPnG1> (need access)
* Get around simple systems by removing spaces between words, alternative spellings/homonyms
* Hate speech = any communication that disparages a person/group on the basis of some characteristic
* Speedups come at cost of speed and scalability – millions of parameters – significant implementation challenge
* Deep learning classifiers– increasing representational power with depth; use different neural structures to learn abstract, high-level features from training data (automated feature extraction – capture hidden patterns/trends); Very Deep Convolutional Neural Networks (VDCNN) (Conneau et al., 2016) (successful in computer vision and text processing) – character-based (increasing depth -> exponentially increasing training time, can’t full test – limitations in computational power), so modified into word-based model, used pre-trained word embedding (FastText) (improves training time, sacrifices no. features); RNNs good at interpreting meaning in text so implemented convolutional bi-directional GRU (Gated Recurrent Unit) w/ attention (like LSTM w/ forget gate and fewer params); PyTorch
* Classical methods – require manual feature engineering (enormous effort, inexhaustive (people can warp messages), always evolving)– logistic regression (baseline – simple to implement), Bayesian models, SVM, random forest classifiers
* Features – word vectors, character n-grams, lexical features, linguistic features (e.g. POS tags), knowledge-based features
* VDCNN classifier – deeper networks can encapsulate more info + have higher test accuracy; characters padded/ truncated to 1024 – look up embedding (size 16) for each char – go through 1D convolutional layer (window size 3, 64 output channels) – pass through 4 convolutional blocks (each w/ 2 1D convolutional layers – doubles output channel) – pooling layer reduces no. embedding by half (downsampling – prevents exponential growth of no. params, consistent memory usage) (*goes through structure and hyperparameters*); trained using CE loss and Adam optimiser; tested 9 and 17 for depths
* Baseline classifier – logistic regression w/ gradient descent update rule; word + character (2-6) n-grams, TF-IDF (measures importance of words in corpus) (TfidfVectorizer from sklearn)
* GRU + LSTM w/ attention (RNN based classifier) – comments padded to max sentence length; used FastText to look up 300-dim vector representations (Mikolov et al., 2018); word embeddings passed through bi-directional GRU/LSTM to obtain sentence embedding – fed into scaled dot produce attention layer (incorporating scale factor into calcs (gives formula)) – scale as value becomes very large with higher dimensions – softmax goes into regions w/ exceptionally small gradients; output 7 classes (clean, toxic, severe toxic, obscene, threat, insult, identity hate) – fully connected linear decoder + softmax for probabilities; trained using CE loss and SGD optimiser
* Using Wikipedia Toxicity Kaggle dataset – split randomly into 80/20 test/dev split to account for overfitting
* Metrics – F1 score, test accuracy
* Best classifiers – bi-LSTM (and GRU) w/ attention and FastText embeddings – pre-trained embeddings boosted accuracy (can calculate subword embeddings; data pre-processing removes tokens like “sucklol” – FastText means can compute embeddings for those tokens; embeddings trained on v. large corpus); attention gives marginal increases in accuracy (ability to prioritize specific parts of sentence over others)
* Logistic regression 11x faster than other classifiers
* Propose cascading classifier – combines multiple models to optimise speed, accuracy – use intermediate steps w/ confidence scores – each step has greater computation cost; test model with logistic regression as 1st step – if output [0.3,0.7] (unsure – 31% of documents) – feeds input into bi-LSTM w/ attention and pretrained model; substantial speedup (22/28 to 5ms latency) and higher accuracy than logistic regression but lower than bi-LSTM (trade-off)
* Future – combine CNNs and RNNs, look at state-of the-art SVMs and feature extraction mechanisms for toxic speech detection

**Paper 11 – “Conversations Gone Awry: Detecting Early Signs of Conversational Failure”**

* Zhang J., Chang J.P., Danescu-Niculescu-Mizil C., Dixon L., Hua Y., Thain N., and Taraborelli D., (2018). “Conversations gone awry: Detecting early signs of conversational failure”, *arXiv preprint arXiv:1805.05345*.
* <https://arxiv.org/pdf/1805.05345.pdf>
* Code and datasets: <https://convokit.infosci.cornell.edu/>
* Detect warning signs (linguistic cues) that civil conversation will be derailed so can salvage conversation
* Humans could tell which conversation more likely to derail 72% of time
* Politeness can shape the course of interactions – soften perceived force of message, act as buffer between conflicting goals, enable parties to save face; politeness tied to social factors: individual status, success of requests/collaborative projects
* Wikipedia dataset (collaborative, goal-oriented setting) – conversations beginning with civil comments and remail healthy/derail into personal attacks; constructed setting that mitigated effects that might trivialise task (e.g. contexts like politics naturally susceptible to antisocial behaviour); 1% of Wikipedia comments contain antisocial behaviour (used Perspective classifier to highlight toxic conversations for dataset – used crowdsourcing to vet labels); only using conversations that start out as civil – look for examples where one of first contributors initiated later attack
* Causal inference used to establish framework focusing on topic-agnostic linguistic cues; pragmatic cues from first comment-reply pair provides signal for direction of conversation; comments prompted by hedged remarks sustain initial civility more than those prompted by forceful questions/direct language
* Inter-annotator agreement – conversation ends in personal attack (67.8%), whether comments in snippet are toxic (87.5%); annotators labelled personal attacks in context of conversations + target; discarded 5% of candidates – crowd identified as not starting out civil; 14% of dataset all annotators agreed had personal attack
* Mitigate topical confounds using matching (for causal inference) – pair awry conversation with on-track conversation if both on same talk page (keep pair closest in time) – 1270 paired conversations over 582 distinct talk pages and 1876 topics – average length of conversation in 4.6 comments.
* Politeness – positive politeness – encourages social connection and rapport – gratitude, greetings, use of ‘please’; negative politeness – dampen imposition on addressee through indirectness/uncertainty; impolite behaviour – direct questions, sentence-starting personal pronouns “Your…”; find by pattern matching on dependency parses of comments
* Rhetorical patterns used to initiate conversations – invitation for working together signals less tension than those that start with statements of dispute; extract rhetorical functions of comments (reflected in type of replies likely to elicit) – derive reply-vectors of phrasings (reflect propensity to co-occur) – perform SVD on term-document matrix of phrasings and replies – derive prompt-vectors (reflect similarities in replies that phrasing prompts) – construct prompt-reply matrix – project into same space as other matrix – clustering yields prompt types (have similar replies) (unsupervised methodology – can be repeated)
* Prompt types (Wikipedia): factual check (statements on article content), moderation (rebukes/disputes on moderations – blocks/reversions, coordination (questions/requests for collaborative editing), casual remark (conversational aside), action statement (explaining/requesting editing action), opinion (on editing challenges/decisions)
* Compute log-odds ratio of marker occurring in initial exchange of awry-turning conversations compared to on-track; in first comment – correspondence between directness and likelihood of future attacks (especially direct questions/start with “You”) – likely to include factual check prompt; on-track conversation start with gratitude/greetings (positive politeness) or coordination – active efforts for constructive teamwork (negative politeness works as well – hedges and opinion prompts – especially in 2nd comment); conversations that derail have more second personal pronouns (contesting initiator) – on-track conversations have more sentence-initial I/We (willingness to work with initiator); attacker-initiated conversations have more direct markers (in attacker and non-attacker – escalating aggression), other conversations – markers of derailment once attacker joins
* Predicting future attacks – extract features from first exchange – logistic regression – report accuracies on leave-one-page-out cross-validation (1 talk page is test data, rest are training)
* Baselines – word count, sentiment lexicon, bag of words - 56.7% accuracy
* Pragmatic features (prompt types + politeness strategies) – 61.6% accuracy
* User features – 51.2% (random chance) – number of edits and anonymity not related to attacker status
* Trained toxicity – Perspective API – 60.5% (trained on additional data from same domain) (64.9% when combined with pragmatic features); majority vote of 3 annotators had 72% accuracy
* Future – look into causal mechanisms of derailments, other domains, more conversational features, predict likelihood of next comment being an attack, identifying pragmatic strategies to bring uncivil conversations back on track

**Other papers on reading list:**

Zampieri M., Malmasi S., Nakov P., Rosenthal S., Farra N., and Kumar R. (2019b). “Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval)”, in *SemEval*.

Hua Y., Danescu-Niculescu-Mizil C., Taraborelli D., Thain N., Sorensen J., and Dixon L., (2018). “Wikiconv: A corpus of the complete conversational history of a large online collaborative community”, *arXiv preprint.*

Nobata C., Tetreault J., Thomas A., Mehdad Y., and Chang Y. (2016). “Abusive language detection in online user content”, in *ICWWW*, pp. 145–153.

Pavlopoulos J., Malakasiotis P., and Androutsopoulos I. (2017b). “Deeper attention to abusive user content moderation”, in *EMNLP*, pp. 1125–1135.

Park J. H. and Fung. P. (2017). “One-step and two-step classification for abusive language detection on twitter”, in *1st Workshop on Abusive Language Online*, pp. 41–45.

Zampieri M., Malmasi S., Nakov P., Rosenthal S., Farra N., and Kumar R.. (2019a). “Predicting the Type and Target of Offensive Posts in Social Media”, in *NAACL.*

Waseem Z., Davidson T., Warmsley D., and Weber I. (2017). “Understanding abuse: A typology of abusive language detection subtasks”, in *1st Workshop on Abusive Langauge Online.*

van Aken B., Risch J., Krestel R., and Löser A. (2018). “Challenges for toxic comment classification: An in-depth error analysis”, in *2nd Workshop on Abusive Language Online*, pp. 33–42.

Davidson T., Warmsley D., Macy M., and Weber I. (2017). “Automated hate speech detection and the problem of offensive language”, in *ICWSM*, pp. 512–515.

Gao L. and Huang R. (2017). “Detecting online hate speech using context aware models”, in *RANLP*, pp. 260–266.

Mikolov T. and Zweig G. (2012). “Context dependent recurrent neural network language model”, in *2012 IEEE Spoken Language Technology Workshop (SLT)*, pp. 234–239. IEEE.

Dixon L., Li J., Sorensen J., Thain N., and Vasserman L. (2018). “Measuring and Mitigating Unintended Bias in Text Classification”, in *Proceedings of AAAI/ACM Conference on Artificial Intelligence, Ethics, and Society.* (metric not robust to variations in class distribution between different identity groups)

Napoles C., Tetreault J., Pappu A., Rosato E., Provenzale B., (2017). “Finding good conversations online: The Yahoo News Annotated Comments Corpus”, in *Proceedings of the 11th Linguistic Annotation Workshop, EACL,* pp. 13–23.

Schmidt A., Wiegand M., (2017). “A survey on hate speech detection using natural language processing”, in *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, pp. 1–10.

Devlin J., Chang M.W., Lee K., and Toutanova K., (2019). “BERT: Pre-training of deep bidirectional transformers for language understanding”, in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4171–4186.

Collobert R. and Weston J., (2008). “A unified architecture for natural language processing: Deep neural networks with multitask learning”, in *Proceedings of the 25th international conference on Machine learning,* pp. 160– 167.

Lucy Vasserman, John Li, CJ Adams, Lucas Dixon. 2018. Unintended bias and names of frequently targeted groups. https://medium.com/the-false-positive/ unintended-bias-and-names-of-frequently-targeted-groups-8e0b81f80a23

**Comparison Models:**

* Jigsaw. 2017. Perspective API. <https://www.perspectiveapi.com/> (open source code at <https://github.com/conversationai/conversationai-models>)
* BERT Base <https://github.com/google-research/bert>
* Constructive comments (Kolhatkar et al. (2020)): <http://moderation.research.sfu.ca/>