**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 - classifier – 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. (2019). “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>
* 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 classifiers 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 remain 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

**Paper 12 – “BioBERT: a pre-trained biomedical language representation model for biomedical text mining”**

* Lee J., Yoon W., Kim S., Kim D., Kim S, Chan H.S., and Kang, J. (2019). “BioBERT: a pre-trained biomedical language representation model for biomedical text mining”, *arXiv preprint arXiv:1901.08746v4*.
* <https://arxiv.org/ftp/arxiv/papers/1901/1901.08746.pdf>
* Modern NLP unsatisfactory – word distribution shift to biomedical corpora
* Tasks: biomedical named entity recognition, biomedical relation extraction, biomedical question answering
* Pre-training BERT on biomedical corpora
* Code: <https://github.com/naver/biobert-pretrained>, <https://github.com/dmis-lab/biobert>
* LSTM and CRF (conditional random field) good performance for named entity recognition, relation extraction and question answering
* Word2Vec – widely known context independent word representation model – previously trained on biomedical corpora
* Training corpora – pubmed and pmc, task specific datasets; trained for 23 days on 8 GPUs
* ELMo, CoVe – context dependent word representations
* BERT – contextualised word-representation model – based on masked language model – pre-trained using bidirectional transformers; needs minimal task-specific architectural modification
* WordPiece tokenisation – mitigates out-of-vocab issue; used original one for BERT
* Metrics – precision, recall, F1 score, strict/lenient accuracy (for question answering), mean reciprocal rank
* Sentence classification for relationship extraction
* 30% questions in BioASQ datasets unanswerable – answers not in given passage
* architecture and pre-training taken from SQuAD for question answering
* Outperformed state-of-the-art models 6/9 times, consistently outperformed BERT

**Paper 13 – “Characterising and Mitigating Aggregation-Bias in Crowdsourced Toxicity Annotations”**

* Balayn A., Mavridis P., Bozzon A., Timmermans B., and Szlávik Z. (2018). “Characterising and mitigating aggregation-bias in crowdsourced toxicity annotations”, in *Proceedings of the 1st Workshop on Subjectivity, Ambiguity and Disagreement in Crowdsourcing, and Short Paper Proceedings of the 1st Workshop on Disentangling the Relation Between Crowdsourcing and Bias Management*, vol. 2276. CEUR.
* <http://pure.tudelft.nl/ws/portalfiles/portal/52000511/paper7.pdf>
* Label aggregation biases results towards certain data samples
* Mitigate majority-bias and get increased prediction accuracy for minority opinions if take into account different annotator labels
* Toxicity subjective – interpreted differently by different people
* Aggregation loses information – decrease in accuracy, unfairness in results
* Using Wikipedia abusive language dataset
* Computed average disagreement rate (ADR) per worker (percentage of annotations different from majority vote)
* Removed annotations of lowest quality workers (spammers) – Worker Quality Score computed with CrowdTruth framework + unit quality score for clarity of sentence
* More low quality workers removed – greater disagreement until stabilises
* Measured accuracy for aggregated and disaggregated data – aggregation testing had higher accuracy but less representative of subjectivity (models biased towards majority opinion)
* Fair algorithm should produce different outputs for same sample depending on reader
* Global metrics don’t inform on bias’s effects – measure sentence and worker-level accuracies on annotations – ambiguity score (% annotation agreement), UQS, ADR, WQS, demographics categories
* Bias mitigation – modify model inputs – remove low-quality workers, feed annotations with worker demographics (continuous/one-hot encoded) – input = (sentence, demographics, annotation)
* Age, gender, education most influencing factors of offensiveness perception
* logistic regression classifier, tf-idf, grid search for hyperparameters
* Bias mitigation – dataset balancing – resample dataset balancing on sentence ambiguity score and majority vote toxicity (bias decreased by equally representing samples with high + low agreement between workers) – removed least frequent annotations in majority vote toxicity and demographics categories – aim for fairness in-between populations
* Models more suited to workers who agree with majority vote – use disaggregated data with adapted ML models
* User representation increases accuracy for workers with high disagreement with majority over using aggregated data/no user model
* Ambiguity-score balanced better (majority vote consensus presence reduced) than demographics-balanced dataset

**Paper 14 – “A Survey on Hate Speech Detection using Natural Language Processing”**

* 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.
* <https://www.aclweb.org/anthology/W17-1101.pdf>
* Hate speech – disparages person/group based on some characteristic
* Basic word filters not enough – hate speech influenced by domain, context and co-occurring media objects, time of posting, world events, identities of author and target
* Toxicity sub-groups: hate speech, abusive, hostile (flames), cyberbullying, insults, profanity, malicious intent, offensive, vulgar, teasing, othering (us-them dichotomy in racist communication)
* Surface features – bag of words (unigrams and n-grams used in most papers – highly predictive), combined with additional features improves performance, character level n-grams attenuate spelling variation problem (more predictive than token n-grams), frequency of URL mentions, punctuation, comment and token lengths, capitalization, words not in English dictionaries, no. non-alpha numeric characters in tokens
* Word generalization – may have data sparsity problem (not in large dataset) (need features in training and testing data), use word clustering, use induced cluster IDs as additional features (Brown clustering algorithm (hard clusters), Latent Dirichlet Allocation (topic distribution for each word – degree word belongs to topic), word embeddings – distributed word representations based on neural networks – for each word vector representation induced from large unlabelled text corpus – different, semantically similar words get similar vectors – replace binary features indicating presence/frequency of words, average vectors of all words in sentence (limited effectiveness), paragraph embeddings – directly represent passage – based on word embeddings – more effective than averaging
* Sentiment analysis – used as auxiliary classification – applied prior to classifier + classifiers that weeds out non-subjective sentences, use as features number of positive, negative, and neutral words (according to sentiment lexicon), hate speech has high degree of negative polarity – SentiStrength predicts polar intensity of utterance
* Word lists – general hate-related terms (gives resources of lists) – some specialised towards particular identity slang terms/slurs, lexicon with good verbs and adjectives, insulting and abusing language dictionary (+ weights for degree of impact – adaptive learning), hate verbs, (not much known about creation of resources) – can employ lexical features in addition to others – contextual factors important – profanity not always hate speech
* Linguistic features – POS-information-enriched tokens (didn’t improve performance significantly), typed dependency relationships – non-consecutive words with relationship captured in 1 feature (significant performance improvement) – chosen using statistical feature selection (Bayesian logistic regression)/manually select relations, offensiveness level score – based on frequency of co-occurrences of offensive terms and user identifiers in same dependency relation, Smokey system features – detect imperatives and co-occurrence of you modified by noun + some semantic features – praise rules – regular expressions using pre-defined good words – politeness rules – polite words/phrases
* Knowledge-based features – use aspects not directly related to language, automatic reasoning over world knowledge – ConceptNet – encodes concepts connected by relations to form assertions (augmented by stereotypes – only works for subtype of hate speech – LGBT)
* Meta-information – background information on author may be predictive (know if written hate speech in past) – no. profane words in history of user, gender (men more likely to post hate speech than women), no. posts, no. replies to post, average replies per follower/region (most not effective for classification), conflicting results on no. comments associated to post
* Multimodal information – can use images/videos as predictive features, use image labels, pixel level image features + captions – predict which images attract hate speech
* Bullying – assign roles to actors involved in event + author (bully, victim, assistant, defender, bystander, reinforcer, reporter, accuser) – predict whether messages directed at author of previous comment/third party, top hate target groups – ethnicity, behaviour, physical characteristics, sexual orientation, class, gender
* Anticipatory governance – notable increases in hate speech in community may indicate future violence/attacks, semantic role labelling and event-based topic extraction, detect tension – sentiment analysis and lexical resources
* Hate speech increases dramatically in hours following terrorist event
* Classifiers – supervised learning – SVMs, deep learning with RNNs, one-step/multi-step (individual classifiers solve subproblems), semi-supervised – bootstrapping – can be used to get additional training data + build lexical resources
* Datasets – authors usually collect and label their own data – no commonly accepted benchmark – Twitter, Instagram, yahoo!, Youtube… - platform demographics different, much fewer hateful than benign comments so have to annotate large number to get enough hateful to balance dataset, not aided by different definitions of toxicity
* Annotations – large differences in agreement in crowdsourcing and expert annotations
* Not enough multilingual approaches

**Paper 15 – “Measuring and Mitigating Unintended Bias in Text Classification”**

* 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.*
* <https://storage.googleapis.com/pub-tools-public-publication-data/pdf/ab50a4205513d19233233dbdbb4d1035d7c8c6c2.pdf>
* Code at: <https://github.com/conversationai/unintended-ml-bias-analysis>
* Pinned AUC metric not robust to variations in class distributions between different identity groups
* Wikipedia Abusive Language Dataset
* False positive bias – non-toxic statements containing certain identity terms given unreasonably high toxicity scores – models over-generalised, overfitting
* Unintended bias – due to disproportionate representation of data with certain identity terms – definition – if model performs better for comments containing some particular identity terms than for comments containing others (identity term bias)
* Fairness tied to demographic equality of model predictions
* Not much work on fairness for text classification tasks – no mitigation strategies
* Gender bias in word embeddings – technique to “de-bias” (Bolukbasi et al., 2016)
* Classifier – CNNs, Keras, TensorFlow
* Group comments for review – publish in batches
* Looked for disproportionate representations for 51 common identity terms – difference between toxic likelihood and overall likelihood
* Relationship between comment length and toxicity – smaller comments more toxic
* Models can capture contextual dependencies – can’t distinguish with insufficient data so over-generalise
* Bias mitigation – add additional data – non-toxic examples of identity terms with most disproportionate data distributions (balanced by length as well) – used unsupervised data from slightly different domain (feasible and effective)
* Test sets – general + identity phrase (synthetic)
* Metrics – AUC – low indicates model performs differently for phrases with different identity terms (may not effectively identify unintended bias on per-group identity dataset – may have low AUC on combined, high within each identity), Equality of Odds (FPR, FNR equal for comments with different identity terms), Error rate equality difference – extent of per-term variation (need score thresholds – used equal error rate threshold), Pinned AUC – threshold-agnostic – detects bias in wider range of use cases – computes AUC on secondary dataset containing sample of comments from identity group and sample from underlying distribution of comments (equally balanced) – capture divergence of model performance on 1 subgroup – want values to be similar across groups, Pinned AUC equality difference – sum of differences between per-term pinned AUC and overall AUC – lower = less variance + bias
* Reduced unintended bias on false positives – didn’t introduce false negative bias

**Paper 16 – “Debiasing Personal Identities in Toxicity Classification”**

* Zorian A.A. and Bikkanur C.S. (2019). “Debiasing Personal Identities in Toxicity Classification”, *arXiv preprint arXiv:1908.05757.*
* <https://arxiv.org/ftp/arxiv/papers/1908/1908.05757.pdf>
* classifiers - TF-IDF w. logistic regression, 2-layer (1 embedding, 2 dense) LSTM with GloVe embeddings (0.1 dropout and input size 50, 2nd dense had 2 sized input and softmax activation (1st had relu)), BERT with ElMo embeddings (best performer – especially in Subgroup AUC) (pre-trained base model, 12 layers, 768 hidden states, 12 heads, 110M parameters, Adam optimizer, batch size 32, learning rate 0.00002) – bertForSequenceClassification class to fine-tune parameters for text classification
* Tokenizer with max no. words set to 10,000 – padded test with max sequence length of 220 (ensure comments equal length)
* Subgroup bias in training set necessary to make accurate classifications – performs poorly when comments containing identities removed from sample
* Substantial work in how unintended bias measured – conversation AI built model to address
* Nuanced metrics – investigate all forms of bias
* CCTK dataset – used toxicity threshold of 0.5
* Metrics – AUC – robust to datasets with unequal numbers of positive and negative examples – threshold agnostic, subgroup AUC (how well model performs on subgroup), BPSN AUC (low = lots of false positives for non-toxic subgroup examples), BNSP AUC (low = lots of false negatives)
* Sarcasm in dataset needs to be addressed – flips polarity of sentence – gender biases in word embeddings

**Paper 17 – “Abusive Language Detection in Online User Content”**

* Nobata C., Tetreault J., Thomas A., Mehdad Y., and Chang Y. (2016). “Abusive language detection in online user content”, in *ICWWW*, pp. 145–153.
* <http://yichang-cs.com/yahoo/WWW16_Abusivedetection.pdf>
* Most commercial methods make use of blacklists and regular expressions
* Noisy data and need for world knowledge
* Keyword spotting leads to false positives, some keywords obfuscated anyway/ever changing slurs and slang terms, context (may be fine to one group, not to another), abusive language can be fluent and grammatical, sentences can have different abusiveness (carry on from previous), sarcasm (requires knowledge of community and potentially users)
* New dataset – 3 annotations per comment + classification on type of toxicity, extracted from comments on Yahoo! Finance and News – moderated by human annotators (at least undergraduate degree + familiar with task + guidelines) – random 10% of all comments sent to moderators + any reported as abusive
* 5-fold cross validation on dataset from other paper
* Features – character n-grams (3-5 char) (model bastardizations of offensive words), token unigrams and bigrams, sentiment, text normalisation, n-grams, linguistic (look for inflammatory words and non-abusive language e.g. politeness or modal verbs – length of comment (in tokens), average length of word, no. punctuations, no. .?” and repeated punctuation, no. 1 letter tokens, no. capitalised letters, no. URLs, no. tokens w. non-alpha characters in middle, no. discourse connectives, no. politeness words, no. modal words (hedging and confidence), no. unknown words, no. insults and blacklist words), syntactic (ClearNLP dependency parser – parent of node, grandparent, POS of parent, POS of grandparent, tuple of word, parent, grandparent, children of node, permutations of word/POS, dependency label connecting word to parent/parent or POS) – capture long range dependencies, and distributional semantics (word + text representations – embedding-derived features – averaging word embeddings of all words in comment/pre-trained embeddings – word2vec to train embeddings from corpora – 200-dimensional embedding vector – embeddings extended to phrases, sentences and paragraphs, entities and documents), pre-processing – transform noise that could impact no. sparse features in model – normalising numbers, replacing long unknown words with same token + repeated punctuation
* Embeddings – learn distributed representations – represented as low-dimensional vectors – jointly learned with distributed vector representations of tokens using distributed memory model, comment embeddings – each comment mapped to unique vector in matrix representing comments (same for words) – comment + word vectors concatenated – predict next word in context, train embeddings of words in comments, using skip-bigram model – window size 10 and hierarchical softmax training, added to other features so low dimensional (10 iterations for efficiency), algorithm not sensitive to comment length – no specific tuning for word weights, needs constant retraining when new comments added so not efficient for online applications (solve with scalable vector tuning and updating for new comments/inferring low-dimensional vector for new comments using gradient descent, parameters, word vectors and softmax weights from trained model/approximating new vector – estimating distance of new comment to previous using words and representations
* Model – learn representation of comments as low-dimensional dense vectors
* [15] S. O. Sood, J. Antin, and E. F. Churchill. Using crowdsourcing to improve profanity detection. In AAAI Spring Symposium: Wisdom of the Crowd, 2012. – first to use crowdsourcing
* Crowdsourcing provides much worse agreement levels than expert moderators
* Combining all features had best performance – character n-grams had largest contribution, syntactic and semantic features didn’t cope well with noise (although comment2vec outperformed word2vec - preserves semantic aspect), averaging embeddings reduced context, word order sensitivity and semantics
* Just using blacklist didn’t perform well – only slightly better when weighted words
* Gold standard references – where all 3 agreed/ when 2 of 3 agreed
* Some false positives dubious – seem abusive but 2/3 rated clean
* Some comments inherently ambiguous
* Having recent data preferable over larger dataset by 5% - can build reasonable model on small set
* Always some amount of unseen words and noise in data – stabilizes as more data added, for certain tasks

**Paper 18 – “Crowd Truth: Harnessing Disagreement in Crowdsourcing a Relation Extraction Gold Standard”**

* Aroyo L. and Welty C. (2013). “Crowd truth: Harnessing disagreement in crowdsourcing a relation extraction gold standard”, *WebSci2013, ACM*.
* <https://www.researchgate.net/profile/Lora_Aroyo/publication/236463327_Crowd_Truth_Harnessing_disagreement_in_crowdsourcing_a_relation_extraction_gold_standard/links/00b7d517f69c26c5d7000000/Crowd-Truth-Harnessing-disagreement-in-crowdsourcing-a-relation-extraction-gold-standard.pdf>
* Usually assumed that single right answer and ground truth quality can be measured in inter-annotator agreement, annotator disagreement reflects semantic ambiguity in target instances – provides useful information
* Propose crowd truth – richer in diversity of perspectives and interpretations and reflects more realistic human knowledge, framework to exploit diverse human responses to annotation tasks – understanding disagreement
* Based on relation extraction from medical text – idea generalizes to other crowdsourced annotation tasks
* Annotator disagreement not noise but signal – sign of vagueness or ambiguity
* Guidelines more brittle as more examples of disagreement arise – classification ends in dissatisfying compromise
* Requirement for high inter-annotator agreements causing task to become overly artificial
* Other papers – disagreement used as trigger for consensus-based annotation, using disagreement to describe set of techniques based on bootstrapping rather than exploiting disagreement, use confusion matrix from disagreement to form similarity cluster
* This paper provides more meaningful feature space for confusion matrix
* Huge gap between expert and lay user’s views – crowd workers could also be spammers
* Worker disagreement = average of cosines between each worker-sentence vector and full sentence vector ( - that worker) – if worker disagrees with a consistently then spammer

**Paper 19 – “Gender Bias in Coreference Resolution: Evaluating and Debiasing Methods”**

* Zhao J., Wang T., Yatskar M., Ordonez V., and Chang K.W. (2018). “Gender bias in coreference resolution: Evaluation and debiasing methods”, *arXiv preprint arXiv:1804.06876*.
* <https://arxiv.org/pdf/1804.06876.pdf>
* WinoBias – benchmark for coreference resolution (find expressions that refer to same entity) focused on gender bias
* Gendered pronouns linked to pro-stereotypical entities with higher accuracy than anti-stereotypical entities
* Social stereotypes in data could impact performance for demographic groups
* Examined 3 different systems – rule-based, feature-rich, end-to-end neural
* Given strong alternative cues, systems can ignore their bias
* Ontonotes 5.0 corpus – female entities significantly underrepresented so replaced male entities by female entities using rule-based approach – anonymise named entities using automatic named entity finder, build dictionary of gendered terms + equivalent for opposite gender using annotators, rules mined by computing word difference between initial and edited sections (she becomes he…), rules applied to all matching tokens
* Word embeddings severely biased – debiased (Bolukbasi et al, 2016)
* Gender lists – counts for how often noun observed in male, female, neutral, plural context – balanced male and female counts for noun phrases
* Implicit human bias can come from imbalanced datasets – under-represented samples neglected
* Can add regularization term that penalizes biased predictions
* Metric – F1 score

**Paper 20 – “Black is to Criminal as Caucasian is to Police: Detecting and Removing Multiclass Bias in Word Embeddings”**

* Manzini T., Lim Y.C., Tsvetkov Y., and Black A.W. (2019). “Black is to criminal as caucasian is to police: Detecting and removing multiclass bias in word embeddings”, *arXiv preprint arXiv:1904.04047*.
* <https://arxiv.org/pdf/1904.04047.pdf>
* Word embeddings trained on stereotyped texts perpetuate and amplify stereotypes, which are amplified by machine learning models
* Remove components that lie in stereotype-related embedding subspaces (for binary labels)
* Multiclass debiasing - train word2vec embeddings using Reddit L2 corpus
* Novel metric for evaluation of bias in collections of word embeddings – mean average cosine similarity (MAC) between word embeddings that should not be related
* Define gender subspace using defining set of words – words in each set represent different ends of bias e.g. (man, woman) – gender subspace computed – compute vector differences of word embeddings of words in each set from set mean, take most significant components of vectors
* Hard debiasing – bias components removed from word not gendered (doctor, nurse…), gendered word embeddings centred and bias components equalised
* Soft debiasing – learn projection of embedding matrix that preserves inner product between biased and debiased embeddings – minimising projection onto bias subspace of embeddings that should be neutral (seeks for linear transformation that minimizes objective)
* Can linearly separate multiple social classes based on components of word embeddings – add additional term for each additional bias class to each defining set, bias subspace given by first k components of PCA evaluation
* Extensive studies of social bias in US – used vocabularies and studies on race to compile lexicons
* Preserves utility of word embeddings

**Paper 21 – “Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings”**

* Bolukbasi T., Chang K.W., Zou J.Y., Saligrama V., and Kalai A.T. (2016). “Man is to computer programmer as woman is to homemaker? debiasing word embeddings”, in *Advances in neural information processing systems*, pp. 4349-4357.
* <https://www.researchgate.net/profile/Venkatesh_Saligrama/publication/305615978_Man_is_to_Computer_Programmer_as_Woman_is_to_Homemaker_Debiasing_Word_Embeddings/links/57a20cd508aeef8f311e0871.pdf>
* Word embedding – popular framework to represent text data as vectors – words with similar semantic meanings have close vectors, vector differences represent relationship between words – trained using second-order methods – require large amounts of data to extract associations and relationships about words
* Trained on Google News corpus
* Gender bias captured by direction in word embedding
* Gender neutral words linearly separable from gender definition words – only remove bias from gender neutral words (direct bias – between gender neutral word and clear gender pair)
* Metrics to quantify direct and indirect gender biases
* Reporting bias – common assumptions often left unsaid
* Bias – different distances between male and nurse and female and nurse – want to be equidistant and maintain embedding utility
* Indirect bias – associations between gender neutral words that clearly arise from gender
* Biases closely aligned with social conception of gender stereotypes
* Low dimensional subspace in embedding captures much of gender bias
* Bias in word morphology (actor vs actress) and number of words (more words sexualise women than men)
* Stereotypes – biases held widely in group, can be complimentary (benevolent sexism) but still harmful (women being associated with attractive may mean less association with professional)
* Certain words predictive of gender (husband for women, baseball for men)
* Fairness through modifying underlying data/classification algorithm
* Metric – normalised cosine similarity
* Amazon Mechanical Turk, at least 5 crowd workers – analysed stereotypes in occupations in word embeddings – dataset stereotypes matched crowd
* Stereotypes prevalent across different embeddings (GloVe on web-crawl corpus as well)
* Analogy generator – seed pair of words determining seed direction (normalised difference between 2 seed words) – want new pair to be parallel to seed direction, words not too far apart (take cos of difference between pairs + distance between new pair) – analogies judged by 10 crowd workers – if made sense + reflected stereotype
* Indirect bias – stereotypically female occupations closer to other stereotypically female occupations (some have legitimate associations – e.g. also sport)
* Gender direction (combine she-he and woman-man) – identify principal components (single direction explains majority of variance, 1st eigenvalue much larger than rest – captures gender subspace) – quantify biases in words and associations – used to predict gender of stereotypes + definitionally gendered words (if closer to she than he), different biases associated with different gender pairs, polysemy (man has multiple meanings), randomness in word counts
* Measure direct bias with metric – many occupation words have substantial component along gender direction, metric for indirect bias too
* Debiasing algorithm – identify direction/subspace of embedding that captures bias; neutralise – gender neutral words 0 in gender subspace, equalise – set of words outside subspace so any neutral word equidistant to both gender terms; soften – reduces differences between sets – maintain as much similarity to original embedding as possible (param for trade-off)
* Gender neutral words – used SVM to determine gender specific words and removed from dataset
* Hard debiasing much more effective than soft debiasing for direct and indirect bias
* Gender biases could capture useful statistics

**Paper 22 – “Mitigating Gender Bias in Natural Language Processing: Literature Review”**

* Sun T., Gaut A., Tang S., Huang Y., ElSherief M., Zhao J., Mirza D., Belding E., Chang K.W., and Wang W.Y. (2019). “Mitigating gender bias in natural language processing: Literature review”, *arXiv preprint arXiv:1906.08976*.
* <https://arxiv.org/ftp/arxiv/papers/1906/1906.08976.pdf>
* Gender bias present in training data, resources, pre-trained models, and algorithms
* Allocation bias – economic issue, system unfairly allocates resources to certain groups over others, models perform better on data associated with majority gender
* Representation bias – systems detract from social identity and representation of certain groups, associations between gender and certain concepts captured in word embedding and model parameters, categories: denigration, stereotyping, recognition, under-representation
* Sentiment analysis systems rank sentences containing female noun phrases to be indicative of anger more often than sentences containing male noun phrases
* Quantifying bias – psychological tests: implicit association test to measure subconscious bias in humans – difference in time and accuracy for humans to categorize words for similar + different concepts (faster to categorize words for males and science rather than arts), word embedding association test – strength of association of concepts – biases in GloVe and Word2Vec embeddings – can be used to track social changes (in female participation in workforce) (same test for sentence encoders e.g. ELMo)
* Gender sub-space in embeddings – as in paper 21 – fails to capture full picture of gender bias in vector spaces – word embeddings with similar biases cluster together, cluster bias – percentage of male/female stereotypical words among k nearest neighbours of word embedding
* Gender-swapping – change gender of gendered nouns, model should perform equally for both sentences – difference in evaluation scores reflects extent of gender bias (False Positive/Negative Equality Distance – metrics)
* Datasets – standard evaluation sets inadequate for measuring gender bias – often contain biases e.g. more male than female references, evaluation may not reveal gender bias, predictions depend on many factors – design datasets to isolate effect of gender of output to probe bias – e.g. WinoBias (may not reflect true distribution but gender bias mistakes avoided), biased datasets may dissuade minorities from using and collecting data – worsens problem
* Coreference resolution – as above
* Debiasing methods – retraining model – address gender bias in early stages/source, costly for resources and time; inference – reduce bias without requiring existence of original training set, patch existing models an adjust outputs (testing-time debiasing)
* Datasets biased towards one gender – augment using gender-swapped version of dataset to balance, apply name-anonymisation (removes gender associations with named entities), identification of gender-specific words - lists by crowd workers; data augmentation flexible and effective at reducing gender bias in models but has larger training times, expensive to create lists, nonsensical sentences created, can lead to inaccurate predictions for some tasks (e.g. Machine translation)
* Gender tagging – add tag indicating gender of source of data to beginning of each data points – prevents from automatically assuming male due to unbalanced datasets, tag parsed separately from rest of data, goal to preserve gender of source, expensive, requires metadata
* Bias fine-tuning – transfer learning from unbiased dataset – ensure model contains minimal bias before fine-tuning model on more biased dataset for target task – models avoid learning biases from training sets and are adequately trained to perform a task, less effective than gender-swapping
* Debiasing word embeddings – blind AI’s perception? Can’t remove gender bias entirely, still cluster bias; remove gender component from gender-neutral words – make gender-neutral words orthogonal to gender direction; learn gender-neutral word embeddings – need to retrain embeddings – isolate gender information in specific dimensions and maintain gender-neutral info in other dimensions – minimise negative difference between gender dimension in gender definitional word embeddings, maximise difference between gender direction and other neutral dimensions (gender dimensions can be used/ignored); don’t work with embeddings in non-Euclidean space – cosine similarity would not apply
* Debiasing algorithms – constrain predictions – constrains optimisation function to ensure predictions fit defined conditions (restrict ratio of males to females predicted for certain activities) Lagrangian relaxation; adjusting discriminator – GAN – generator learns with respect to protected gender attribute (prevents discriminator from identifying gender) – generalisable to gradient-based learning models
* Not clear that methods generalise/how techniques combine for ideally unbiased system/may introduce noise – performance degradation, hand-crafted techniques may encode implicit bias of developers

**Other papers on reading list:**

Pavlopoulos J., Malakasiotis P., and Androutsopoulos I. (2017b). “Deeper attention to abusive user content moderation”, in *EMNLP*, pp. 1125–1135. <https://www.aclweb.org/anthology/D17-1117.pdf>

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. <https://arxiv.org/pdf/1810.04805.pdf>

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*. <https://arxiv.org/pdf/1903.08983.pdf>

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.* <https://arxiv.org/pdf/1810.13181.pdf>

Park J. H. and Fung. P. (2017). “One-step and two-step classification for abusve language detection on twitter”, in *1st Workshop on Abusive Language Online*, pp. 41–45. <https://arxiv.org/ftp/arxiv/papers/1706/1706.01206.pdf>

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.* <https://arxiv.org/pdf/1902.09666.pdf>

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.* [*https://arxiv.org/pdf/1705.09899.pdf*](https://arxiv.org/pdf/1705.09899.pdf)

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. <https://arxiv.org/pdf/1809.07572.pdf>

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. <https://arxiv.org/pdf/1703.04009.pdf>

Gao L. and Huang R. (2017). “Detecting online hate speech using context aware models”, in *RANLP*, pp. 260–266. <https://arxiv.org/pdf/1710.07395.pdf>

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. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.258.5120&rep=rep1&type=pdf>

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. <https://www.aclweb.org/anthology/W17-0802.pdf>

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. <https://thetalkingmachines.com/sites/default/files/2018-12/unified_nlp.pdf>

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

“Detecting hate speech on World Wide Web” <https://www.aclweb.org/anthology/W12-2103.pdf>

“Limitations of Pinned AUC for Measuring Unintended Bias” <https://arxiv.org/pdf/1903.02088.pdf>

“Correlating Self-Report and Trace Data Measures of Incivility: A Proof of Concept” <https://chrisjvargo.com/wp-content/uploads/2018/12/1Final%20Proof.pdf>

**Useful resources:**

* Jigsaw. 2017. Perspective API. <https://www.perspectiveapi.com/> (open source code at <https://github.com/conversationai/conversationai-models>)
* https://github.com/conversationai/unintended-ml-bias-analysis
* BERT Base <https://github.com/google-research/bert>
* Constructive comments (Kolhatkar et al. (2020)): <http://moderation.research.sfu.ca/>
* Blacklist/hate speech vocabulary - <https://hatebase.org/>