AI TACTICS, TECHNIQUES AND PROCEDURE TO AVOID HARM FOR TEXTUAL QUERIES IN ALGORITHMIC MODELS

ARSLAN, AYSEKOK

Oxford Alumni of Northern California

IJASR 2022
VOLUME 5
ISSUE 3 MAY – JUNE **ISSN: 2581-7876**

Abstract: Driven by recent developments in AI, Web search queries such as auto-suggestion have become very popular. It start with a discussion of methods and techniques used for recommender systems and suggests a deployment approach with continuous iteration to reduce associated risks.A process based on the use-cases for a recommender system API to achieve respectful behavior is also explored. The paper concludes that as there is no silver bullet for AI, in addition to the control mechanisms that should be put into place to ensure transparency in collection, use and dissemination of the AI data, there should also be codes and standards in place to ensure users' privacy and to raise awareness about the AI so that individuals can protect themselves.

Keywords: ML, AI, big data, IOT

Introduction

Recommender systems such asWeb search queries orauto-suggestion (often also referred to as autocompletion) are becoming more popular given the proliferation of AI into our daily lives.Existing research has rarely considered data sparsity, cold boot and data noise.

Given the fact that in real scenarios, a resource selection system has to compute a huge amount of data, resource optimization is currently an urgent problem to be solved. Therefore, designing and implementing a method for selecting content resources will be of great significance to the research on recommender systems. This study aims to fulfill this gap.

Review of Existing Work

Algorithm recommendation refers to tracking users' network behavior, using some mathematical algorithms to calculate personal characteristics, environmental characteristics and other relevant information, and infer the content that users may like. The application of algorithm recommendation mainly involves three levels:

- First, the content level includes subject category, content source and channel, service function, interactive topic discussion and so on.
- The second is the user level, including the user's region, interest, occupation, age, gender and educational level.
- The third is the contextual information involving the time and place.

Below is a description of main types of algorithms forrecommender systems:

- *Demographically-basedrecommendation*: This is the simplest recommendation algorithm, which uses the user's basic information, such as age, gender, location. The aim is to analyze the similarity of users and recommend the items preferred by similar users to this user.

- *Content-based recommendation*:This recommendation algorithm uses the similarity of the item itself rather than the similarity of the user as the basis.

- *Collaborative filtering basedrecommendation*:The demographics-based mechanism only considers the characteristics of the user, while the user-based collaborative filtering mechanism calculates the similarity of

the user on the data of the user's historical preferences. The basic assumption is that users who like similar items may have the same or similar tastes and preferences.

- Item-based Collaborative Filtering Recommendation: The difference between item-based collaborative filtering recommendation and content-based recommendation is also the difference in similarity calculation methods. The former is inferred from the user's historical preference, while the latter is based on the attribute characteristic information of the item itself.

- Model-based Collaborative Filtering Recommendation: Model-based collaborative filtering recommendation is to construct a recommendation model based on sample user preference information, and then predict recommendations based on real-time user preference information.

- *Rule-based recommendation*:The core of the algorithm recommendation based on association rules is to mine out the association rules among products. In content channel applications, program recommendation based on user clicks, thumb up number and subscription are all recommendation based on association rules. In addition, the platform can predict users' preferences based on the correlation between their historical data and program characteristics.

- *Hybrid algorithm recommendation*:Multiple recommendation algorithms are combined to avoid or compensate for the weaknesses of each recommendation technique. By analyzing users' playing, subscribing, commenting and other operation behaviors, the algorithm generates data and gives priority to more content in line with users' interests and hobbies for recommendation.

Understanding which suggestions should be construed as problematic and how to efficiently detect them also requires examining possible dimensions including,

1) *content* (Olteanu, *et al.*, 2020; Miller and Record, 2017; Yenala, *et al.*, 2017);
2) *targets* (Olteanu, *et al.*, 2020; Olteanu, *et al.*, 2018; UN Women, 2013);
3) *structure* (Santos, *et al.*, 2017); and,
4) *harms* (Miller and Record, 2017).

A query suggestion may constitute harmful speech if the query could be perceived as hateful, as it offends, shows a derogatory attitude; or if the query appears related to *e.g.*, defamatory content promoting negative, unproven associations, or statements about individuals, groups, organizations.

Large scale recommender systems can be trained in highly parallel and distributed training environments, with a large amount of randomness in training the models..It is also not uncommon for the metrics measured to change from design and testing to deployment, bringing into question the utility of the experimental testing phase.

Lack of replicability, where researchers are unable to reproduce published results with a given model, has been identified as a challenge in the field of machine learning (ML). Irreproducibility is a related but more elusive problem, where multiple instances of a given model are trained on the same data under identical training conditions, but yield different results. An ML model attempts to learn the best model parameters that fit the training data by minimizing a loss, which can be imagined as a landscape with peaks and valleys, where the lowest point attains an optimal solution. For deep models, the landscape may consist of many such peaks and valleys. The activation function used by the model governs the shape of this landscape and how the model navigates it.

In real systems, two models that are supposedly the same, may behave very differently when deployed in production. For some applications this may be acceptable. However, for some recommendation systems, providing different recommendations for the same request is undesirable, specifically in cases where a typical user would expect equal results. Generalizing such results to deep learning, with highly non-convex loss landscape, is even more challenging.

Irreproducibilty is exacerbated if the current recommendations and their user engagements become the training data for future recommendations, as in reinforcement learning systems. Divergence of the training data gradually expands model and future recommendation differences. Practical systems often retrain models and redeploy new

versions, use continuous (online) training, update models with fresh new data, or replace models by new generations.

Irreproducibility affects not only the user, but also the engineering development cycle. Model developers usually use aggregate prediction accuracy metrics on either validation or progressivevalidation [8] (in online models) data to judge experimental models. With huge training datasets, training is expensive, timely, and resource intensive. Without taking reproducibility into consideration, a model can be deployed in experimental stage, show favorable metrics, but then, when retrained to be deployed in production, produce unfavorable results. This can be expensive to diagnose and address.

Many factors contribute to irreproducibility in deep models [13, 18, 19, 44, 48, 49, 56]. The highly non-convex objective [18], combined with nondeterminism in training [49] and underspecification [13] of over-parameterized deep networks, can lead training models to optima at different locations in a manifold or sets of optima. Nondeterminism can emerge from the highly parallelized, highly distributed training pipelines, quantization errors, hardware types [56] and more. Slight deviations early in training due to these can lead to very different models [1]). Nondeterminism in training from random initialization, parallelism, distributed training, data shuffling, quantization errors, hardware types, and more, combined with objectives with multiple local optima contribute to the problem of irreproducibility. Some of these factors, such as initialization, can be controlled, but it is impractical to control others. Optimization trajectories can diverge early in training by following training examples in the order seen, leading to very different models. Several recently published solutions [1, 2, 3] based on advanced combinations of ensembling, self-ensembling, and distillation can mitigate the problem, but usually at the cost of accuracy and increased complexity, maintenance and improvement costs.

Moreover, in a production development cycle, new models, architectures, and algorithms are being developed. A new model may not always be aligned with a previous generation, and it may not be possible to warm start its now different parameters to those of a previous model. With huge datasets in practical systems, enforcing determinism [35] in training is also not an option.

A huge set of input features are learned, but every training example consists only of an insignificant fraction of the feature set. Those features are mapped to embedding vectors that are concatenated as the input to the deep network. Embeddings constitute the dominating fraction of model parameters. For keeping equal complexity, a network component of an ensemble will have narrower embedding vectors and hidden layers than a single network counterpart. Thus in a production systemwith limited parameter complexity, using ensembles may improvereproducability, but at the expense of accuracy (which directly effects important downstream metrics). Ensembles may make production models more complex and harder to maintain. Production models may include various parts with special handling of various special cases.

Networks with smooth activations (e.g., GELU, Swish and Softplus) can be substantially more reproducible. They may exhibit a similar objective landscape, but with fewer regions, giving a model fewer opportunities to diverge. The Rectified Linear Unit (ReLU) activation [36] has been instrumental to deep networks in recent years. With backpropagation it gives simple updates, accompanied with superior accuracy. Due to its non-smoothness, ReLU imposes an extremely non-convex objective surface. With such surface, the order in which updates are applied is a dominant factor in determining the optimization trajectory, providing a recipe for irreproducibility.

Compression of deep networks into smaller networks that attempt to describe the same information is the emerging area of distillation [21]. Predictions of a strong teacher train a weaker student model. The student is then deployed. This approach is very common if there are ample training resources, but deployment is limited, as for mobile network devices. Co-distillation [4] (see also [54]) embraces training ensembles and distillation to address irreproducibility. Instead of unidirectional transfer of knowledge, several models in an ensemble distill information between each other, attempting to agree on a solution. The method requires more training resources to co-train models, but deployment only requires a single model (which can be an ensemble by itself).

Other recent approaches to address the irreproducibility problem attempted to anchor the a solution to some constraint [7, 45] but also degrade performance by constraining the model's ability to converge to a better solution.

A recommendation set can be a stream of past recommendations that a user engaged with, or it can be an outcome set provided for a specific query issued by the user. A recommended item is assigned a label based on the user response or engagement with the item. In the simplest binary case, which we focus on, the label is either positive, i.e., the user engaged with the item, or negative, the user did not engage with the item. Many such systems are in the sparse regime. There is a huge selection of items the user can engage with, and a huge set of features that can describe the request as well as the items themselves. Features can be properties of the request, the item, the combination of the two, the user, the recommendation user interface, the recommendation rendering, and more.

Recommender deep neural networks can be fully connected or of other architectures. Due to the extreme sparsity, models can train mainly on individual item engagement label rates, although other losses can also be introduced for various purposes. Numerical input features act as inputs to the network. Categorical features are mapped into embedding vectors, each vector representing a category. All features/embeddings are concatenated into an input layer of the deep network. The embeddings constitute the majority of the model parameters, but as described, only an insignificant portion of the stored embedding vectors is present in any training example.

Understanding reproducibility, especially in deep networks, where objectives are not convex, is an open problem. Nevertheless, Zhang and Kong (2020) emphasized that there are a number of recommendation methods used for different purposes. Li (2021) adopted a hybrid recommendation algorithm incorporating big data -a personalized recommendation algorithm based on collaborative filtering recommendation algorithm (CF), which obtained the user evaluation matrix based on big data by using the Pearson correlation coefficient to calculate the similarity between users and form the nearest neighbor set, and to generate the user-based recommendation set.

ML models have been found useful in improving the detection rate of problematic queries but at the expense of increased false positive rates (P. Gupta & Santos, 2017).While ML models avoid the effort required to hand-craft rules, they require human effort to annotate collections of queries (both problematic and non-problematic) in order to train models.In addition to this, AI methods such as Backpropagation, Support Vector Regression, Gradient Boosting Classifier, Bayesian Classifier, Artificial Neural Network, and Decision Tree can also be employed. The latter involves a mix of advanced statistical methods and AI heuristics.

Large scale systems can be trained in highly parallel and distributed training environments, with a large amount of randomness in training the models. While some systems may tolerate such randomness leading to models that differ from one another every time a model retrains, for many applications, reproducible models are required, where slight changes in training do not lead to drastic differences in the model learned.

When determining whether a suggestion is problematic, the potential for various harmful effects — along with their severity, frequency or impact (Boyarskaya, *et al.,* 2020) — should also be factored in (e.g., discomfort versus physical harm).Techniques that have been explored for this purpose include gradient boosting decision trees (Chuklin and Lavrentyeva, 2013), long short-term memory networks (Yenala, et al., 2017), and the deep structured semantic model (P. Gupta and Santos, 2017).

Classification is the most applied approach while the Decision Tree classifier is the most common algorithm (Mohamad and Tasir, 2019). Decision Tree classifiers are frequently used as they are easy to understand and have high predictive accuracy. Furthermore, Neural Network, Bayesian Networks, Rule Induction, and Support Vector Machines are examples of classification techniques that are frequently used to perform prediction.

Decision Tree classifiers
Naïve Bayes classifiers
Random Forest
SVM classifiers
K-Nearest Neighbours

Table 1. Classification of Algorithms

Other popular algorithms are Bayesian Networks and Random Forest. Bayesian Networks are graphical models with nodes and directed edges that are probabilistic in nature.

In recent years, federated learning has become a popular machine learning paradigm, especially for recommender systems. In federated learning, a group of clients cooperate to train a global model without uploading local datasets. Each client can only access its data, which protects the privacy of participants' data in the training. The original purpose of federated learning is to collaborate with all participants to obtain an aggregation model [3].One of the methods for federated learning is knowledge distillation.The basic idea of knowledge distillation is to take the output of an extensive complex network and transmit it as knowledge to a small network [9]. In the training process, the small network can learn the information of the real labels of the data and can learn the relationship between different labels and can then be converted into a compact network. but also learns from the experiences of other networks to further improve the generalization ability of the model.

Although these models have achieved near universal state of the art across thousands of tasks, the downside is that they require significant number of task-specific training examples to finetune the model. Additionally, at least a portion of the model parameters must be updated to fit the task, adding complexity from model finetuning and deployment.

Broadly, modeling refers to approaches for predicting either the next token in a sequence or for predicting masked spans (Devlin et al., 2019b; Raffel et al., 2020). Predictable power-laws of model quality through scaling the amount of data, parameters, and computation have made this a reliable approach for increasingly more capable models (Kaplan et al., 2020).

The improvements in these models have primarily come from one or more of the following approaches: (1) scaling the size of the models in both depth and width; (2) increasing the number of tokens that the model was trained on; (3) training on cleaner datasets from more diverse sources; and (4) increasing model capacity without increasing the computational cost through sparsely activated modules.

Demonstrating that prompting the model to generate explicit inference chains can drastically increase the quality of the predictions themselves. In other words, the model's generation (rather than just understanding) capabilities can be immensely beneficial even for tasks that are modeled as categorical prediction or regression, which typically do not require significant data generation.

Needless to say, thereare still many open questions about the ideal network architecture and training scheme for future generations of models. The goal should be to explore a diverse array of novel architectural choices and training schemes, and combine the most promising systems with the extreme scaling capabilities to develop a large-scale, modularized system that will have broad generalization capabilities across multiple modalities.

Implementation Framework

Regardless of the methods and techniques used, the deployment approach should emphasize continuous iteration, and make use of the following strategies aimed at maximizing the benefits of deployment while reducing associated risks:

- Pre-deployment risk analysis, leveraging a growing set of safety evaluations and red teaming tools
- Starting with a small user base
- Studying the results of pilots of novel use cases (e.g., exploring the conditions under which we could safely enable longform content generation, working with a small number of customers)
- Implementing processes that help keep a pulse on usage (e.g., review of use cases, token quotas, and rate limits)
- Conducting detailed retrospective reviews (e.g., of safety incidents and major deployments)

A process based on the use-cases for a recommender system API to achieve respectful behavior would entail in general following steps:

Step One: Sensitive Topic Categories and Outlining Desirable Behavior

Engineers should select categories that they prioritize as having direct impact on human well-being and describe desired behavior in the form of following categories. It should be noted that the following list is not exhaustive and prioritization depends on context.

- *Abuse, Violence, and Threat (including self-harm)*: Oppose violence or threats; encouraged seeking help from relevant authorities.
- *Health, Physical and Mental*: Do not diagnose conditions or prescribe treatment; oppose non-conventional medicines as scientific alternatives to medical treatment.
- *Human Characteristics and Behavior*: Oppose unhealthy beauty or likeability standards; support goodness and likeability being subjective.
- *Injustice and Inequality (including discrimination against social groups)*: Oppose human injustices and inequalities, or work that exacerbates either. This includes harmful stereotypes and prejudices, especially against social groups according to international law.

Step Two: Crafting the Dataset and Fine-Tuning

Developers can craft a values-targeted dataset of various samples; in a question-answer format and then fine-tuned their models on this dataset using standard fine-tuning tools.

In order to inform appropriate policy interventions, the dataset should also take other modalities into account. For example, developers initially focused on long form text generation as a threat vector, given prior cases of influence operations that involved people manually writing long form misleading content. Given that emphasis, they set maximum output lengths for generated text. Yet, output restrictions had little effect on policy violations—so short-form content amplifying or increasing engagement on misleading content could offer a greater risk.
Examples of limitations in existing datasets include the following:

- an overly narrow focus (e.g., just measuring occupational bias),
- an overly broad focus (e.g., measuring all under the umbrella of "toxicity"),
- a tendency to abstract away the specifics of use and context,
- a failure to measure the *generative* dimension of real world model use (e.g., using multiple choice style),
- prompts that differ stylistically from those typically used in real world model use cases,
- not capturing dimensions of safety that are important in practice (e.g., an output following or ignoring a safety-motivated constraint in the instruction), or
- not capturing types of outputs to be correlated with misuse (e.g., harmful content).

Step Three: Evaluating Models

Developers then can use quantitative and qualitative metrics: human evaluations to rate adherence to predetermined values; toxicity scoring which could also not capture all nuance in toxicity and host their own biases.

There are some limitations to the methods in use such as classifier-based data filtration. For instance, operationally defining the content areas to detect via filtration is challenging and filtration itself can introduce harmful biases. Additionally, the labeling of toxic data is a critical component and ensuring the mental health of these labelers is an industry-wide challenge.

While aligned models have practical advantages such as reducing the need for "prompt engineering" (providing examples of the desired behavior to steer the model in the right direction), saving space in the model's context window which can be used for other purposes and other models in a way that reduces risks of harm has posed various technical and policy challenges.

To overcome such challenges, no-code AI platforms are also being utilized. As ML requires knowledge of programming languages and the expert of data scientists to create and test, no-code AI platforms can be used to create and deploy ML models by performing the necessary preprocessing steps of choosing and extracting features,

and creating and comparing a range of different ML models through an easy-to-use graphical user interface (Bardoliwalla, 2022). The ultimate aim is to validate and ensure that these models are quite accurate and make sense from an interpretability perspective (Bardoliwalla, 2022).

This evolution of such ML tools has reached a level where non-coders and less technical people can perform most of their data querying tasks through easy-to-use graphical tools and without the assistance of expert data analysts. This evolutionary trend enabled AI and ML to provide every person with the ability to predict what is going to happen (Bardoliwalla, 2022). In other words, AI and ML could be put into the hands of millions of individuals to make them deliver AI- and ML-specific outcomes. According to some scholars, the AI Cloud will evolve into an end-to-end no-code platform that covers the entire ML development lifecycle (Bardoliwalla, 2022).

Various data engineering tools for gathering, segmenting, labeling, updating, and managing the datasets can be used to train and validate ML models to overcome AI hazards. There are also opportunities to further streamline the automated ML process by continually monitoring the accuracy of not only the model in production, but also challenger models that can potentially replace the main ML model as context and conditions change (Bardoliwalla, 2022). Such no-code environments would allow for more functionality and be manifested in a few simple clicks inside of a GUI.

Discussion

Many of the issues covered throughout the paper are complex and difficult to mitigate. Some may even argue that these issues are so intractable that the safest approach is to disable the autosuggest feature entirely (despite its known benefits to users), or to at least allow users to opt-in to the feature with a warning about its potential problems.

Alternatively, the feature could be used only to surface prior frequent queries made by the current user, which might preserve some benefit while eliminating problematic suggestions learned from other users. Nevertheless, the goal of this study is to review on-going existing issues posed by the search autosuggestion feature, with the hope that highlighting them will inspire new research and development efforts into the challenging aspects of the problems, both technical and social.

As Sharma, Kawachi, and Bozkurt (2019) state, in addition to the control mechanisms that should be put into place to ensure transparency in collection, use and dissemination of the AI data, there should also be codes and standards in place to ensure users' privacy and to raise awareness about the AI so that individuals can protect themselves.

Conclusion

AI has a mixed reputation. There is no silver bullet for responsible deployment, so everyone in the process should try to learn about and address models' limitations, and potential avenues for misuse, at every stage of development and deployment in order to learn as much as possible about safety and policy issues at small scale and to incorporate those insights prior to launching larger-scale deployments.

Last, but not least, when it comes to designing for AI, the ultimate aim should be to promote human creativity, responsibility, sustainability, and social connectedness as well as to increase self-efficacy, bring joy, spread compassion, and respect human dignity.

REFERENCES

1. A. Akhtar, 2016. "Google defends its search engine against charges it favors Clinton," USA Today (10 June) https://www.usatoday.com/story/tech/news/2016/06/10/google-says-search-isntbiased-toward-hillary-clinton/85725014/, accessed 14 July 2020.
2. W. Arentz and B. Olstad, 2004. "Classifying offensive sites based on image content," Computer Vision and Image Understanding, volume 94, numbers 1–3, pp 295–310.doi: https://doi.org/10.1016/j.cviu.2003.10.007, accessed 14 July 2020.

3. K. Arnold, K. Chauncey, and K. Gajos, 2018. "Sentiment bias in predictive text recommendations results in biased writing," GI '18: Proceedings of the 44th Graphics Interface Conference, pp. 42–49.doi: https://doi.org/10.20380/GI2018.07, accessed 14 July 2020.
4. BBC, 2019. "Microsoft Word AI 'to improve writing'" (7 May), at https://www.bbc.com/news/technology-48185607, accessed 14 July 2020.
5. F. Cai and M. de Rijke, 2016. "A survey of query auto completion in information retrieval," Foundations and Trends in Information Retrieval, volume 10, number 4, pp. 273–263, and at https://www.nowpublishers.com/article/Details/INR-055, accessed 14 July 2020. doi: http://dx.doi.org/10.1561/1500000055, accessed 30 January 2022.
6. N. Diakopoulos, 2014. "Algorithmic accountability reporting: On the investigation of black boxes" (3 December), at https://www.cjr.org/tow_center_reports/algorithmic_accountability_on_the_investigation_of_black_boxes.php, accessed 14 July 2020.
7. M. Golebiewski and d. boyd, 2018. "Data voids: Where missing data can easily be exploited," Data & Society, at https://datasociety.net/wp-content/uploads/2018/05/Data_Society_Data_Voids_Final_3.pdf, accessed 14 July 2020.
8. A. Gulli, 2013. "A deeper look at Autosuggest," Microsoft Bing Blogs (25 March), at https://blogs.bing.com/search/2013/03/25/a-deeper-look-at-autosuggest/, accessed 14 July 2020.
9. P. Gupta and J. Santos, 2017. "Learning to classify inappropriate query-completions," In: In: J. Jose, C. Hauff, I. SengorAltngovde, D. Song, D. Albakour, S. Watt, and J. Tait (editors). Advances in information retrieval. Lecture Notes in Computer Science, volume 10193. Cham, Switzerland: Springer, pp 548–554.
doi: https://doi.org/10.1007/978-3-319-56608-5_47, accessed 14 July 2020.
10. S. Karapapa and M. Borghi, 2015. "Search engine liability for autocomplete suggestions: Personality, privacy and the power of the algorithm," International Journal of Law and Information Technology, volume 23, number 3, pp. 261–289.
doi: https://doi.org/10.1093/ijlit/eav009, accessed 30 January 2022.
11. P. Lee, S. Hui, and A. Fong, 2002. "Neural networks for Web content filtering," IEEE Intelligent Systems, volume 17, number 5, pp. 48–57.
doi: https://doi.org/10.1109/MIS.2002.1039832, accessed 14 July 2020.
12. R. Magu, R., K. Joshi, and J. Luo, 2017. "Detecting the hate code on social media." arXiv:1703.05443v1 (16 March), at https://arxiv.org/abs/1703.05443, accessed 14 July 2020.
13. K. McGuffie and A. Newhouse, 2020. "The radicalization risks of GPT-3 and advanced neural language models," arXiv:2009.06807v1 (15 September), at https://arxiv.org/abs/2009.06807, accessed 9 April 2021.
14. D. Metaxa-Kakavouli and N. Torres-Echeverry, 2017. "Googles role in spreading fake news and misinformation," Stanford Law School, Law and Policy Lab, Fake News & Misinformation Policy Practicum (31 October), at https://www-cdn.law.stanford.edu/wp-content/uploads/2017/11/SSRN-id3062984.pdf, accessed 30 January 2022.
15. B. Miller and I. Record, 2017. "Responsible epistemic technologies: A social-epistemological analysis of autocompleted Web search," New Media & Society, volume 19, number 12, pp. 1,945–1,963.
doi: https://doi.org/10.1177/1461444816644805, accessed 14 July 2020.
16. A. Olteanu, C. Castillo, F. Diaz, and E. Kcman, 2019. "Social data: Biases, methodological pitfalls, and ethical boundaries," Frontiers in Big Data (11 July).doi: https://doi.org/10.3389/fdata.2019.00013, accessed 14 July 2020.
17. A. Olteanu, C. Castillo, J. Boy, and K. Varshey, 2018. "The effect of extremist violence on hateful speech online," Proceedings of the Twelfth International AAAI Conference on Web and Social Media, at https://www.aaai.org/ocs/index.php/ICWSM/ICWSM18/paper/view/17908/17013, accessed 14 July 2020.
18. A. Olteanu, K. Talamadupula, and K. Varshney, 2017. "The limits of abstract evaluation metrics: The case of hate speech detection," WebSci '17: Proceedings of the 2017 ACM on Web Science Conference, pp. 405–406.doi: https://doi.org/10.1145/3091478.3098871, accessed 30 January 2022.
19. Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnil, 2014. "Learning semantic representations using convolutional neural networks for Web search," WWW '14 Companion: Proceedings of the 23rd International Conference on World Wide Web, pp. 373–374.doi: https://doi.org/10.1145/2567948.2577348, accessed 14 July 2020.
20. H. Yenala, M. Chinnakotla, and J. Goyal, 2017. "Convolutional bi-directional LSTM for detecting inappropriate query suggestions in Web search," In: J. Kim, K. Shim, L. Cao, J.G. Lee, X. Lin, and Y.S. Moon (editors). Advances in knowledge discovery and data mining. Lecture Notes in Computer Science, volume 10234. Cham, Switzerland: Springer, pp. 3–21. 16.doi: https://doi.org/10.1007/978-3-319-57454-7_1, accessed 14 July 2020.