Current Opinion
CO 3 (4), 354– 361 (2023)
Received 23 May 2023 | **Revised 22 June** 2023 | **Accepted** 13 **July** 2023 | **Online Available 10 August** 2023
https://doi.org/10.52845/currentopinion.v3i4.231

Current Opinion
Current Opinion

OPEN ACCESS JOURNAL **ISSN(O) 2795- 935X**

Original Article

Exploring Use of Chatgpt for Integrated Design Environments: A Quality Model for Chatgpt

AyseKok Arslan

Researcher Silicon, Valley Oxford Northern California Alumni San Francisco, Northern California, United States

Abstract:
Given the proliferation of ChatGPT into our every day lives, the aim of this paper is to present a quality design framework for integrating the use of ChatGPT in development environments.
The study starts with a discussion of integrated design environments (IDEs) and explores its recent for generative AI models. It also highlights commonly used techniques in AIGC, and addresses concerns surrounding trustworthiness and responsibility in the field.
Finally, it explores open problems and future directions for the use of this quality framework, highlighting potential avenues for innovation and progress.

Introduction

The use of ChatGPT has proliferated into almost all aspects of our every day lives from healthcare to education. Within this regard, the aim of this paper is to present a brief review on existing IDEs in order to propose a quantitative and qualitative metrics for ChatGPT to be integrated into the existing development environments.

This evaluation is highly significant since it will allow software developers to broaden their perspective regarding which characteristics and quality metrics should be covered by the use of ChatGPT.

2 Overview of Integrated Design Environments

Software engineering is an engineering discipline whose goal is the cost-effective development of software systems [1, 2]. There are various techniques, approaches, programming paradigms, and tools for software development and automatic source code generation.

IDE (Integrated Development Environment) is the most basic tool which is used by software maintainers to perform system maintenance activities. IDE's combine the features of many tools in one package. IDEs may, for example, be used for the development of enterprise-level applications [4].

Some scholars already described the research performed to analyze the requirements for the development of an IDE for embedded system design [2]. The work concluded with a recommended approach for the development of an IDE for embedded system design.

Some researchers [5] stated that software development could be eased with an IDE, which allows for using different individual tools on one single development platform. Unfortunately, when developing software for a particular embedded system, the development of an IDE for a certain device can be expensive, since the development of an IDE requires a lot of resources.

Software Engineering literature has proposed several works related to software development evaluation. There are two main approaches: quantitative (objective) evaluations and qualitative (subjective) evaluations.

On the one hand, according to [46], quantitative evaluations are based on identifying the effects of using a tool in measurable terms. On the other hand, qualitative evaluations — also known as feature analyses — are based on identifying the requirements that the user possesses to perform a particular task/activity and on linking these requirements to the tool's features that can support the task or tasks.

This assessment method describes each of the three aspects that constitute the needs of a software developer; it also describes the features that a tool for software development, especially for source code generation, must possess in order to satisfy each of the identified needs.

Quality in use is defined as the ability of the software product to enable users to achieve specific goals with effectiveness, productivity, safety, and for each of the three aspects, the final score will be the highest score assigned by a member of the evaluation team composed of two software engineers, two graphic designers, and two software developers.

As a quality framework, ISO 9241-11 [54] defines interoperability as the ability of the software product to interact with one or more specified systems. The interoperability factor defines whether the software can be easily combined to enhance its capabilities.

ISO 9241-11 [54] defines usability as the extent to which a product can be used by specific users in order to achieve specific goals with effectiveness, efficiency, and satisfaction in a specific context.

Conventional software design may be best accomplished with a "divide and conquer" process; but in cases where integrative expertise is needed, collaboration may require sharing key lower-level details (and not all). As AI modelling and development now requires fusing human needs within technical designs, points of intense collaboration across expertise roles will occur.

Selbst and colleagues [10] argue that social context information may be critical for some design considerations, so standard abstraction methods may require alteration for AI design. To design AI systems with fairness in mind, teams need to collaboratively define fair performance by considering diverse stakeholders, contexts of use, and assessment criteria (i.e., disaggregated evaluation [4]) which can also influence quality metrics.

The metrics that are proposed for this evaluation focus on the measurement of the quality of an automatic code generation tool to generate source code. Therefore, they are based on internal metrics defined in the ISO/IEC 9126 standard.

- Task time: The estimated time of work spent to develop a basic application.
- Help frequency: The frequency of use of the help and/or documentation tools.
- Modification complexity: The estimated work time spent on changing data sources of the applications already generated.
- Inline documentation completeness: Ratio of the number of scripts, functions, or variables having documentation to the number of implemented scripts, functions, or variables.

As far as data security is concerned, interoperability refers to the ability of a software tool to protect data in order to avoid unauthorized persons or systems to read or modify this data.

Moreover, documentation should contain information on the operations of the software system. However, it can also be aimed at end users with the purpose of facilitating the interaction between the end users and the tool (e.g., a training manual).

The next section explores important models such as foundation and generative models in AIGC before proceeding with the suggested quality model.

3 Background of Ai Models

AIGC refers to content that is generated using advanced Generative AI (GAI) techniques, as opposed to being created by human authors, which can automate the creation of large amounts of content in a short amount of time.

Technically, AIGC refers to, given human instructions which could help teach and guide the model to complete the task, utilizing GAI algorithms to generate content that satisfies the instruction.

Generally, GAI models can be categorized into two types: unimodal models and multimodal models (Fig. 1.0). Unimodal models receive instructions from the same modality as the generated content modality, whereas multimodal models accept cross-modal instructions and produce results of different modalities.

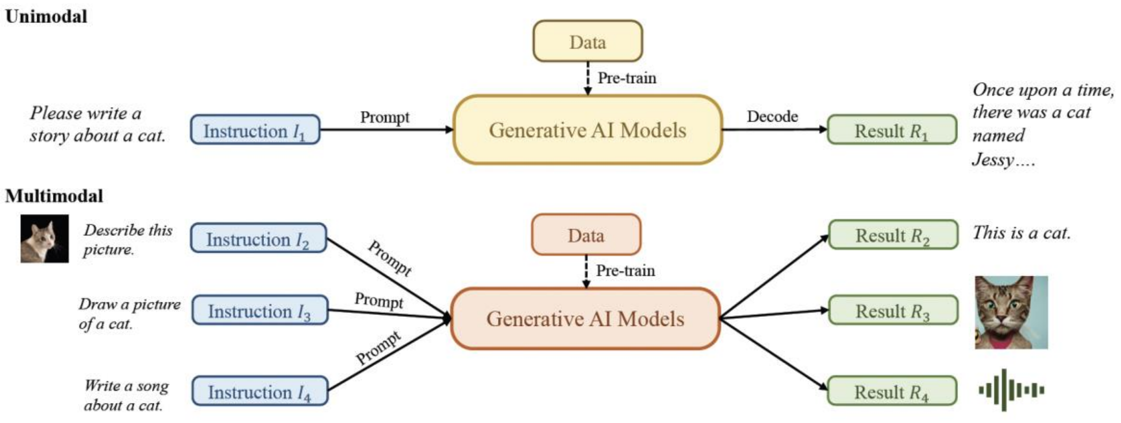


Figure 1. Overview of AIGC (Credit: Yenala et al (2019))

By combining these advancements, models have made significant progress in AIGC tasks and have been adopted in various industries, including art [14], advertising [15], and education [16]. The next section looks into these models in more detail.

3.1 Foundation Model

Transformer is the backbone architecture for many state-of-the-art models and is mainly based on a self-attention mechanism that allows the model to attend to different parts in an input sequence. Transformer consists of an encoder and a decoder. The encoder takes in the input sequence and generates hidden representations, while the decoder takes in the hidden representation and generates output sequence.

Generally, these transformer based pre-trained language models can be commonly

classified into two types based on their training tasks: autoregressive language modeling and masked language modeling [41].

Given a sentence, which is composed of several tokens, the objective of masked language modeling, e.g., BERT [42] and RoBERTa [43], refers to predicting the probability of a masked token given context information. The most notable example of masked language modeling is BERT [42], which includes masked language modeling and next sentence prediction tasks.

Despite being trained on large-scale data, the AIGC may not always produce output that aligns with the user's intent, which includes considerations of usefulness and truthfulness. In order to better align AIGC output with human preferences, reinforcement learning from human feedback (RLHF) has been applied to fine-tune models in various applications such as Sparrow, InstructGPT,

and ChatGPT [10, 46]. Typically, the whole pipeline of RLHF includes the following three steps: pre-training, reward learning, and fine-tuning with reinforcement learning.

3.2 Generative Models

Generative models have a long history in artificial intelligence, dating back to the 1950s with the development of Hidden Markov Models (HMMs) [20] and Gaussian Mixture Models (GMMs) [21]. These models generated sequential data such as speech and time series. However, it wasn't until the advent of deep learning that generative models saw significant improvements in performance.

In early years of deep generative models, different areas do not have much overlap in general. In natural language processing (NLP), a traditional method to generate sentences is to learn word distribution using N-gram language modeling [22] and then search for the best sequence. However, this method cannot effectively adapt to long sentences. To solve this problem, recurrent neural networks (RNNs) [23] were later introduced for language modeling tasks , allowing for modeling relatively long dependency.

This was followed by the development of Long Short-Term Memory (LSTM) [24] and Gated Recurrent Unit (GRU) [25], which leveraged gating mechanism to control memory during training. These methods are capable of attending to around 200 tokens in a sample [26], which marks a significant improvement compared to N-gram language models.

In recent years, researchers have also begun to introduce new techniques based on these models. For instance, in NLP, instead of fine-tuning, people sometimes prefer few-shot prompting [38], which refers to including a few examples selected from the dataset in the prompt, to help the model better understand task requirements. In visual language, researchers often combine modality-specific models with self-supervised contrastive learning objectives to provide more robust representations.

3.2.1 Unimodal Models

Generative language models (GLMs) are a type of NLP models that are trained to generate readable human language based on patterns and structures in input data that they have been exposed to. These models can be used for a wide range of NLP tasks such as dialogue systems [58], translation [59] and question answering [60].

Recently, the use of pre-trained language models has emerged as the prevailing technique in the domain of NLP. Generally, current state-of-the-art pre-trained language models could be categorized as masked language models (encoders), autoregressive language models (decoders) and encoder-decoder language models.

Decoder models are widely used for text generation, while encoder models are mainly applied to classification tasks. By combining the strengths of both structures, encoder-decoder models can leverage both context information

and autoregressive properties to improve performance across a variety of tasks.

3.2.2 Multimodal Models

The goal of multimodal generation is to learn a model that generates raw modalities by learning the multimodal connection and interaction from data [7]. This connection and interaction between modalities can sometimes be very intricate, which makes the multimodal representation space hard to learn compared to the unimodal one.

Vision Language Generation: The encoder-decoder architecture is a widely used framework for solving unimodal generation problems in computer vision and natural language processing. The encoder is responsible for learning a contextualized representation of the input data, while the decoder is used to generate raw modalities that reflect cross-modal interactions, structure, and coherence in the representation.

Vision Language Encoders: Recently, the development of encoders for single modalities has advanced significantly, leading to the question of how to learn contextualized representations from multiple modalities. A common way to do this is to combine modality-specific encoders using a fusion function and then leverage multiple pre-training tasks to align the representation space [37, 134, 135].

4 Suggested Quality Framework

As seen in Figure 2.0, when it comes to engineering for AI, the software engineering approach should reflect the human mental models as accurate as possible. Within an AI-driven approach, human mental-models target the following:

(1) understanding how end-users would perform a task on their own and the challenges they might face; that is, the task model;

(2) understanding people’s expectations about what the AI should do, and setting expectations for people about AI behavior, which is referred to as the expectation model, and

(3) identifying the best type of AI interaction experience given the situational context; namely, the interaction model.

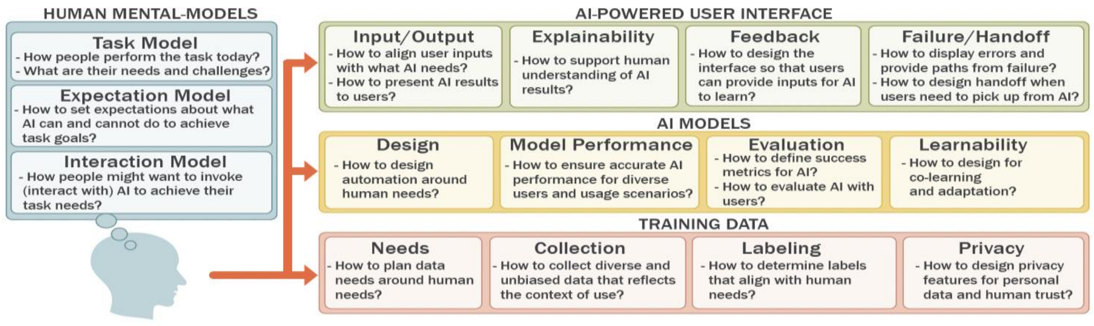
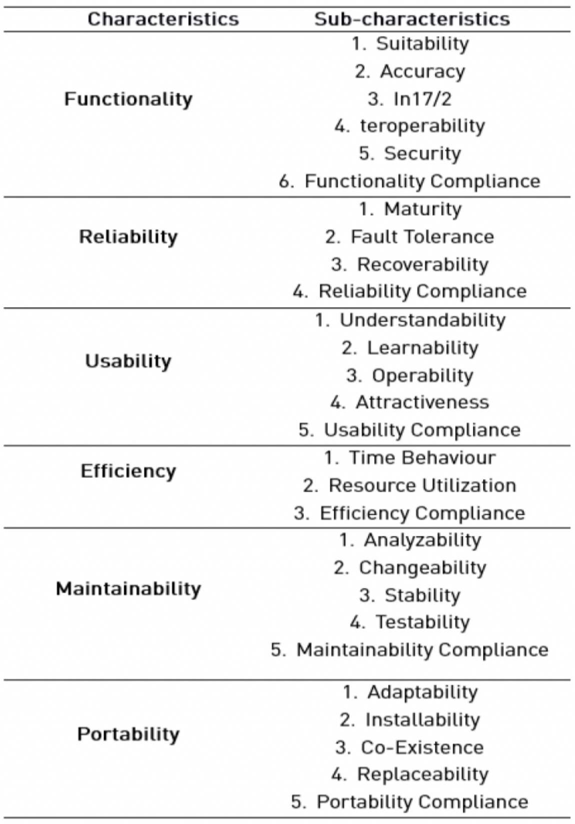

HUMAN MENTAL-MODELS
Task Model
- How people perform the task today?
- What are their needs and challenges?
Expectation Model
- How to set expectations about what AI can and cannot do to achieve task goals?
Interaction Model
- How people might want to invoke (interact with) AI to achieve their task needs?
AI-POWERED USER INTERFACE
Input/Output
- How to align user inputs with what AI needs?
- How to present AI results to users?
Explainability
- How to support human understanding of AI results?
Feedback
- How to design the interface so that users can provide inputs for AI to learn?
Failure/Handoff
- How to display errors and provide paths from failure?
- How to design handoff when users need to pick up from AI?
AI MODELS
Design
- How to design automation around human needs?
Model Performance
- How to ensure accurate AI performance for diverse users and usage scenarios?
Evaluation
- How to define success metrics for AI?
- How to evaluate AI with users?
Learnability
- How to design for co-learning and adaptation?
TRAINING DATA
Needs
- How to plan data needs around human needs?
Collection
- How to collect diverse and unbiased data that reflects the context of use?
Labeling
- How to determine labels that align with human needs?
Privacy
- How to design privacy features for personal data and human trust?

Figure 2. Overview of AI development steps

Within this regard, the suggested quality model for integrating ChatGPT takes ISO/IEC 9126-1 [45] as a reference point to maintain software quality in a structured set of characteristics and sub-characteristics. The characteristics are (Fig. 3 Quality Model):

- Functionality: A set of attributes that bear on the existence of a set of functions and their specified properties. The functions are those that satisfy explicit or implied needs [45].
- Reliability: A set of attributes that bear on the capability of software for maintaining its level of performance under stated conditions for a stated period of time [45].
- Usability: A set of attributes that bear on the effort needed for use and the individual assessment of such use by a stated or implied set of users [45].
- Efficiency: A set of attributes that bear on the relationship between the level of performance of the software and the amount of resources used under stated conditions [45].
- Maintainability: A set of attributes that bear on the effort needed to make specified modifications [45].
- Portability: A set of attributes that bear on the ability of software to be transferred from one environment to another [45].



Characteristics	Sub-characteristics
Functionality	1. Suitability 2. Accuracy 3. In17/2 4. teroperability 5. Security 6. Functionality Compliance
Reliability	1. Maturity 2. Fault Tolerance 3. Recoverability 4. Reliability Compliance
Usability	1. Understandability 2. Learnability 3. Operability 4. Attractiveness 5. Usability Compliance
Efficiency	1. Time Behaviour 2. Resource Utilization 3. Efficiency Compliance
Maintainability	1. Analyzability 2. Changeability 3. Stability 4. Testability 5. Maintainability Compliance
Portability	1. Adaptability 2. Installability 3. Co-Existence 4. Replaceability 5. Portability Compliance

Figure 3 Overview of a ChatGPT Quality Model

This assessment method has a long tradition within software engineering and information systems literature [5]; moreover, it has been used for other quality evaluations [11, 12].

Each qualitative aspect has a score based on a 3-point Likert scale [9]. The aspects are presented in the following text. The scale used for the measurement of the identified aspects is a 3-point Likert scale [9] in which "3" represents the best score and "1" represents the worst score as it is represented below:

• 3 points: strongly addressed (S.A.)

• 2 points: partially addressed (P.A.)

• 1 point: not addressed (N.A.)

The overall evaluation for each software tool will be the sum of the final scores in the three aspects.

5 Recommendations

While AIGC has the potential to be incredibly useful in many different applications, it also raises

significant concerns about security and privacy. This section aims to provide some recommendations.

5.1 Security

Factuality: Systematic definitions of truthfulness standards and approaches for governing AI-generated content were proposed in Truthful AI [24]. The standard proposed by Truthful AI aims to avoid "negligent falsehoods" and explicitly train AI systems to be truthful via curated datasets and human interaction.

Based on GPT-3, WebGPT [25] proposed a humanoid prototype that models the AI answering process into web searching and evidence-composing phrases. Since the model is trained to cite its sources, the factual accuracy of AI-generated content is significantly improved in multiple benchmark datasets [26, 27].

Toxicity: Besides utility, it is important for AI-generated content (AIGC) to be helpful, harmless, unbiased, and non-toxic. Extensive research has been conducted on the potential harm caused by deployed models [229–231], which can include biased outputs [232, 233], stereotypes [234], and misinformation [25].

To address this issue of toxicity in the language domain, OpenAI proposes InstructGPT [10], which aligns language models with human preferences by using human feedback as a reward signal to fine-tune the models, ensuring more relevant and safe responses. Concurrently, Google proposes LaMDA [26], a family of neural language models specialized for safe and factual dialog by leveraging fine-tuning and external knowledge sources.

5.2 Privacy

Membership inference: The goal of the membership inference attack (MIA) is to determine whether

an image x belongs to the set of training data. Wu et al. [238] investigated the membership leakage in

text-to-image (diffusion-based and sequence-to-sequence-based) generation models under realistic

black-box settings. Specifically, three kinds of intuitions including quality, reconstruction error,

and faithfulness are considered to design the attack algorithms.

Data Extraction: The objective of a data extraction attack is to retrieve an image from the set of training data, denoted as $x \in D$. The attack can be considered a success if the attacker is able to obtain an image $\hat{x}$ that closely resembles image $x \in D$.

Compared to the membership inference attack, the data extraction attack poses stronger privacy risks to the model. The feasibility of such an attack might be due to the memorization property of large-scale models [243], in which they turn to memorize parts of their training data.

6 Open Problems and Future Directions

Many fundamental challenges to developing a high quality model capable of performing well in real world

applications still exist. For example, it is now increasingly well-understood that large language models trained on

unlabeled datasets will learn to imitate patterns and biases inherent in their training sets [10]. Such biases can be hard to detect since they manifest in a wide variety of subtle ways. For example, the axes of marginalization differ greatly across geo-cultural contexts, and how they manifest in pre-trained language models is an under-studied area [11].

Known approaches to mitigate undesirable statistical biases in generative language models include attempts to filter

pre-training data, train separate filtering models, create control codes to condition generation, and fine-tuning models. While these efforts are important, it is critical to also consider the downstream applications

and the socio-technical ecosystems where they will be deployed when measuring the impact of these efforts in mitigating harm. For example, bias mitigations in certain contexts might have counter-intuitive impacts in other geo-cultural contexts [10].

The field of algorithmic bias measurement and mitigation is still growing and evolving rapidly, so it will be important to continue to explore novel avenues of research to ensure the safety of dialog agents Future work should explore the benefits of greater coordination across the research community and civil society in the creation of benchmarks and canonical evaluation datasets to test for harmful and unsafe content.

It should also be taken into account that various traits measured for safety objectives depend heavily on socio-cultural contexts. Therefore, any meaningful measure of safety should take into account the societal context where the system will be used, employing a “participatory finetuning” approach that brings relevant communities into the human-centered data collection and curation processes.

Another challenge in GAI models relates to reasoning which is a crucial component of human intelligence that enables us to draw inferences, make decisions, and solve complex problems. However, even trained with large scale dataset, sometimes GAI models could still fail at common sense reasoning tasks [256, 257]. Recently, more and more researchers began to focus on this problem.

Model training is always limited by compute budget, available dataset and model size. As the size of pretraining models increases, the time and resources required for training also increases significantly. This poses a challenge for researchers and organizations that seek to utilize large-scale pretraining for various tasks, such as natural language understanding, computer vision, and speech recognition.

Another issue pertains to the efficacy of pretraining with large-scale datasets, which may not yield optimal results if experimental hyperparameters, such as model size and data volume, are not thoughtfully designed. As such, suboptimal hyperparameters can result in wasteful resource consumption and the failure to achieve desired outcomes through further training.

Overall, while AI-generated content holds significant promise in various domains, it is crucial to address these concerns to ensure that its use is responsible and beneficial for society as a whole.

7 Conclusion

This paper has presented a quality framework for integrating CHATGPT into the development environment. These kinds of reviews and subsequent tools assessments are extremely important issues for software developers to identify the characteristics and functionalities that cover use of ChatGPTs.

A future study could expand the evaluation of tools by including other qualitative and quantitative characteristics in the evaluation process. Some features that could be considered are correctness of the generated code, required computer resources, and integration with other tools.

References

1. Akhtar: 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-isnt biased-toward-hillaryclinton/85725014/, accessed 14 July 2020 (2016)
2. Arentz, W and B. Olstad: 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. (2016).
3. Gulli, A: 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. (2013)

4. McGuffie and A. Newhouse: 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. (2020)
5. Miller and I. Record, M: 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. (2017)
6. Olteanu, C. Castillo, J. Boy, and K. Varshey: 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 (2018)
7. Olteanu, K. Talamadupula, and K. Varshney: 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. (2017)
8. Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnil: 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 2017.
9. Olteanu, C. Castillo, F. Diaz, and E. Kcman: 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. (2019)
10. H. Yenala, M. Chinnakotla, and J. Goyal: 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. 3–16.doi: https://doi.org/10.1007/978-3-319-57454- 7_1, accessed 14 July 2020. (2017)