Human perceiving behavior modeling in evaluation of code generation models

Sergey Kovalchuk, Vadim Lomshakov, Artem Aliev
Huawei
{sergey.kovalchuk,vadim.lomshakov,artem.aliev}@huawei.com

Abstract

In this study, we evaluated a series of code generation models based on CodeGen and GPTNeo to compare the metric-based performance and human evaluation. For a deeper analysis of human perceiving within the evaluation procedure, we implemented a 5-level Likert scale assessment of the model output using a perceiving model based on the Theory of Planned Behavior (TPB). Through this analysis, we demonstrated an extension of model assessment as well as a deeper understanding of the quality and applicability of generated code for answering practical questions. The approach was evaluated with several model settings in order to assess diversity in the quality and style of answer. With the TPB-based model, we showed a different level of perceiving of the model result, namely, personal understanding, agreement level, and readiness to use the particular code. With this analysis, we investigate a series of issues in code generation, namely, natural language generation (NLG) problems observed in the context of programming and question-answering with code.

1 Introduction

Recent advances in natural language generation (NLG) support rapid growth in potential areas of application. One of the significant successes of NLG is the possible generation of code in various settings (Lu et al., 2021; Zhong et al., 2022): the translation of explicit specification into code, fixing errors, suggesting short snippets, etc. The common practice in NLG problems is the usage of known metrics (BertScore, BLEU, etc.) to evaluate models. Moreover, specific metrics dedicated to code generation evaluation were developed, such as CodeBLEU (Ren et al., 2020), RUBY (Tran et al., 2019), and others. Still, recent studies (Evtikhiev et al., 2022) show that the direct application of metrics often leads to issues in code generation evaluation.

With this in mind, investigated the applicability of human evaluation widely spread in NLG problems (De Mattei et al., 2021; Hämäläinen & Alnajjar, 2021) to assess an alternative approach to code evaluation and a deeper understanding of human perceiving of code generation (and NLG output in general). The study poses two research questions. First, how is human perceiving reflected by the text- or code-oriented NLP metrics? Second, what is the structure of human perceiving in the human evaluation procedure in the question-answering scenario? Here we consider perceiving as an act of becoming subjectively aware and conscious of the observed information.

The structure of the paper is as follows. The next section describes the datasets used in the study. The following section presets the details of code generation models' selection and preparation. Section 4 describes human evaluation solutions and procedures. Section 5 discusses the study results and evaluation results. Finally, Sections 6 and 7 provide a discussion and concluding remarks respectively.

2 Dataset

Within the study, we focused on question answering (QA) with a generation of short snippets as answers to real-world problems such as questions asked in Stack Overflow[1] (SO). For consistency, we added the following restrictions to the questions and answers considered within the study, taking SO as a reference for the analysis.

[1] https://stackoverflow.com/

Proceedings of the 2nd Workshop on Natural Language Generation, Evaluation, and Metrics (GEM), pages 287 - 294
December 7, 2022

- We considered “conceptual” and “API usage” questions according to the taxonomy presented in (Beyer et al., 2020).
- We selected the questions that mainly contain a short textual description without explicit code presented.
- Contrarily, we use answers with explicit code snippets giving the solution to the proposed problem.
- To further specify the scope of the study, we only considered one programming language, Python, as it is one of the most popular ones.

To prepare an appropriate dataset we followed two steps. First, we used the publicly available dataset CoNaLa[2] (Yin et al., 2018) with explicitly identified train (2379 entries) and test parts (500 entries). The dataset originates from SO and contains explicit short questions and reference code snippets.

Alternatively, we prepared our own dataset from the original data on SO questions available on Stack Exchange[3]. To follow our requirements, we selected questions with the tag “python”. For reference, we selected answers that earned maximum scores according to the SO data. To filter questions on presence or absence of code, we search the text for <pre><code> in HTML data. After that, we selected questions with no explicit code paired with answers with a single code snippet. Finally, we used regular expressions following (Beyer et al., 2020) to select “conceptual” and “API usage” classes of questions. Furthermore, we performed cleaning of the code (e.g. removing decorations inserted by software (“>>>”, “In [1]:”, etc.), comments, and checked the parsing status using Tree-sitter[4]. After these steps, we obtained a dataset containing 42292 entries (pairs of questions and answers). Out of them, we selected 1000 entries as a test dataset. The test dataset was built using questions from 2021 and beyond to lower possible data leaking as we are using models trained on publicly available data.

3 Code generation models

3.1 Model selection

To select models for our study, we evaluated those which are publicly available, computationally inexpensive, and applicable on our data with finetuning. First, we selected GPT-Neo(-J)[5] (Black, Sid et al., 2021, p.), which shows high performance compared to Codex (Xu et al., 2022). Second, we chose CodeGen-mono-2B by Salesforce (Nijkamp et al., 2022), which was trained not only on the Pile dataset but also separately on the code from BigQuery and BigPython. CodeGen-mono-2B shows good results on HumanEval, which was similar to the Codex model of the same size (Chen et al., 2021). Additionally, we picked CoPilot by Microsoft as an industrial SOTA reference solution, which is also based on Codex (Chen et al., 2021).

3.2 Finetuning

Finetuning was performed for the selected models on training datasets from both CoNaLa (further denoted as FT:C) and SO (further denoted as FT:SO). We used Transformers and DeepSpeed libraries with common hyperparameters (optimizer = AdamW, Adam betas = (0.9, 0.999), Adam epsilon = 1e-08, weight decay = 0.0, learning rate = 5e-06, learning rate decay = linear, batch size(#samples) = 40, fp16). Moreover, we performed experiments with various code wrapping for prompt preparation. A query wrapped as a multiline comment to the generated code was selected as a well-performing baseline. Additionally, we performed a series of experiments in prompt engineering and selected the best performing solution for further experiments (denoted as FT:C+).

4 Human evaluation

4.1 User interface implementation

For performing the human evaluation, a user interface (UI) was developed as a web application using the Dash[6] framework (see Figure 1). The UI enables the collection of feedback information in two ways.

First (***HF1*** - human feedback 1), the UI shows a pair of answers generated for the same question by different models. The pairwise comparison of two answers was collected from the user by asking them to select the best answer with a 3 or 5 (depending on configuration) levels Likert scale

[2] https://conala-corpus.github.io/
[3] https://stackexchange.com/
[4] https://tree-sitter.github.io/
[5] https://github.com/EleutherAI/gpt-neo
[6] https://dash.plotly.com/

from -2 (the left answer is the best) to +2 (the right answer is the best). This feedback is collected for further research purposes to improve model performance with human-centered prediction models (currently considered as future research plans following the works (Nakano et al., 2022; Stiennon et al., 2020)).

Second (***HF2***), the user was asked to assess both answers (code snippets) with three scores using a 5-level Likert scale (from -2 to +2) by estimating:

- The general consistency of the code (whether the code is readable/understandable). The scale is considered to reflect how well the user ***understands*** the answer.
- The correctness of the answer with respect to the proposed question. The scale is considered to reflect the user's ***agreement*** with the answer.
- The usability of the provided answer. The scale is considered to reflect the user's expected ***intention to use***.

These scales analyze human behavior aspects by assessing the information perceiving in alignment with a model based on the theory of planned behavior (TPB) (Ajzen, 1991), widely used to quantify human behavior as reflected by attitude, subjective norms, and perceived control affecting target intention to use a considered technology. In our case, we consider "agreement" as the first criterion, reflecting the general user's attitude, "understand" as the second one, reflecting correspondence to subjective norms and perceived control, and "use" as a final target criterion, the obtained intention to use the solution.

The human feedback is collected for each pair of answers storing the scores provided by the user and his/her provided name. After each evaluation round (ended by clicking the "Submit" button), the interface is updated with a new pair of answers for further analysis.

For evaluation purposes, the original and finetuned models were applied to test sets in the selected datasets (CoNaLa and SO) forming a collection of alternative answers to 1500 questions. Applying the selected models and filtering empty answers, we obtained 10013 answers of different origin and quality. On each round, a random sample of two different answers was presented to the user. With this approach, a subset of 1364 questions was selected, supported with two or more answers by different models.

4.2 Human perceiving assessment

Within the presented study, we focused on the internal structure of answer perceiving during human evaluation. Thus, we performed a deeper analysis of HF2 to understand the connections between different features. For this purpose, we consider three main groups of features.

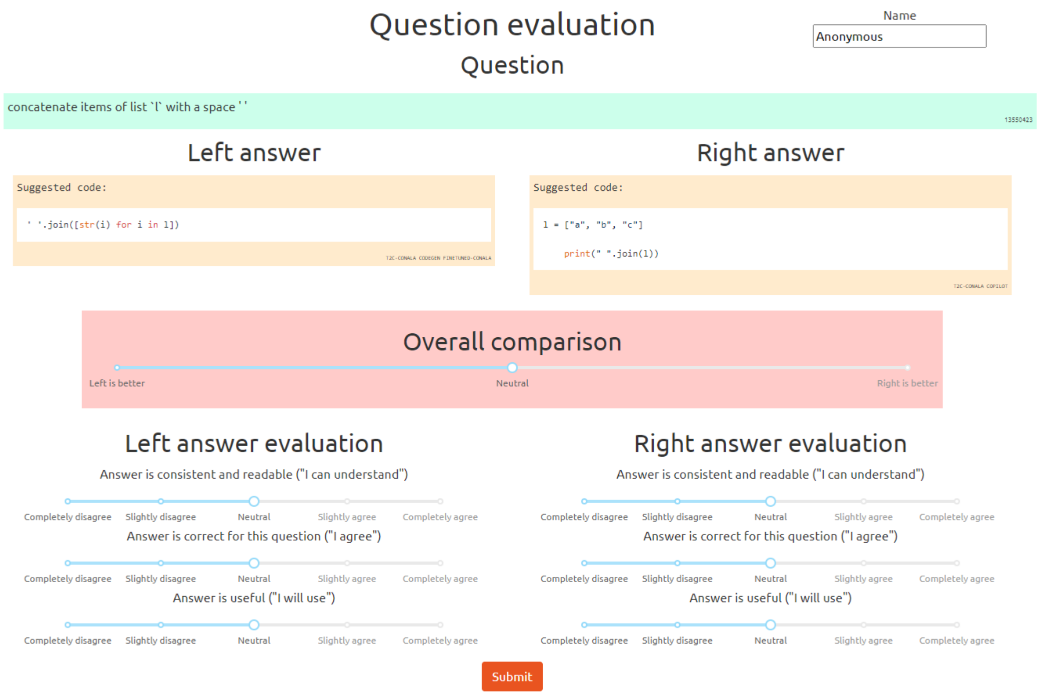

Figure 1. User interface for human evaluation

	BertScore		Rouge		CodeBLEU		Ruby		SacreBLEU	
Model	**mean**	**std**	**mean**	**std**	**mean**	**std**	**mean**	**std**	**mean**	**std**
Dataset: CoNaLa										
CodeGen	0.8068	0.1827	0.3142	0.2638	0.2821	0.2379	0.2791	0.2363	0.1196	0.1493
CodeGen *FT:C*	0.9017	0.1132	0.5532	0.2976	0.4848	0.2861	0.5392	0.3151	0.2142	0.1759
CodeGen *FT:SO*	0.8326	0.0379	0.1707	0.1316	0.0918	0.1095	0.1014	0.1348	0.0522	0.0658
CodeGen *FT:C+*	**0.9235**	0.0511	**0.6070**	0.2763	**0.5802**	0.2519	**0.6246**	0.2845	**0.2571**	0.1952
GPT-Neo	0.7370	0.1688	0.0503	0.0688	0.0785	0.0670	0.1460	0.1190	0.0111	0.0151
GPT-Neo *FT:C*	0.8366	0.1518	0.2926	0.2648	0.2298	0.2401	0.2619	0.2759	0.0900	0.1096
GPT-Neo *FT:SO*	0.8251	0.0380	0.1453	0.1122	0.0749	0.0858	0.0909	0.1192	0.0400	0.0455
CoPilot	0.8520	0.0398	0.3668	0.2075	0.2815	0.1554	0.2812	0.1899	0.1179	0.1162
Dataset: SO										
CodeGen	0.7434	0.1838	0.0790	0.0951	0.1687	0.1549	0.0974	0.1025	0.0176	0.0331
CodeGen *FT:C*	**0.8178**	0.0923	**0.1473**	0.1447	**0.3311**	0.2488	**0.1615**	0.1935	**0.0396**	0.0572
CodeGen *FT:SO*	0.8099	0.0825	0.1298	0.1188	0.1729	0.1773	0.0933	0.1062	0.0327	0.0452
GPT-Neo	0.7543	0.1286	0.0511	0.0577	0.1411	0.1074	0.0991	0.1101	0.0093	0.0123
GPT-Neo *FT:C*	0.7912	0.1488	0.1193	0.1265	0.2986	0.2295	0.1433	0.1606	0.0292	0.0419
GPT-Neo *FT:SO*	0.8003	0.1126	0.1156	0.1109	0.1574	0.1675	0.0953	0.1069	0.0275	0.0378

Table 1. Metric-based evaluation

FG1 (feature group 1) includes the common metrics used for the NLG task. The selection of metrics includes general purpose metrics (BertScore, Rouge, SacreBLEU) and metrics specific to code generation problems (CodeBLEU, Ruby). The metrics were evaluated for each model applied for test datasets.

FG2 includes votes collected from the users along three selected scores, namely, subjective consistency (understanding), subjective correctness (agreement), and subjective intention (use).

FG3 includes simple test features (linguistic features), namely, question and answer length, average lines number in answer, and average lines number in question.

To answer the proposed research questions, we analyzed the interconnection between the features in the three groups by assessing the pairwise mutual information (MI) between features. As we focused mainly on perceiving structure and interconnection, the main analysis and interpretation were applied to a) internal MI between features in FG2; b) MI between features in FG2 and features in other groups.

5 Results

5.1 Metric-based evaluation

Weused a common train-eval-test split for evaluation. In the case of the CoNaLa dataset, the test dataset was pre-selected by the authors. In the case of the SO dataset, we composed the test dataset as random samples of 1000 answers dated 2021 and beyond, while the answers dated 2020 and earlier were used for training. In both cases, we split the training dataset into train and validation parts as 9:1 randomly.

Table 1 shows the main evaluation results according to the selected NLP metrics. In the case of the CoNaLa dataset, the best results were obtained by CodeGen FT:C+, followed by CoPilot. In the case of the SO dataset, the best performance was achieved with CodeGen FT:C.

5.2 Human evaluation

With the implemented UI and judgment collection procedure, the human evaluation was performed in a semi-open way by exposing the UI with a pre-defined collection of answers to independent groups of users of different backgrouds, but having a basic understanding of coding and the principles of software engineering. The user set includes MSc/PhD students and researchers in computer science and related areas, as well as professional software developers. The diversity in experience of the users enables further deeper analysis of the nature of human perceiving , along with the possible assessment of experience influence.

During a week-long evaluation period, the collection of votes for 614 answers in the HF2 part was collected from 43 different users. Within the analysis, we exclude the original models and only consider finetuned models. The average scores for FG2 with the selected pairs model/dataset are presented in Table 2.

We observe the highest perceiving in CoPilot and finetuned CodeGen. The CoNaLa dataset shows slightly lower Understand scores compared to the SO dataset. At the same time, the SO dataset shows negative Agree and Use scores reflecting wrong (but consistent) answers generated by the model. A more interesting observation is the diversity in scores exhibited by the best CodeGen and CoPilot in the CoNaLa dataset: CoPilot shows slightly higher Understand and Agree scores, while the Use score is much higher for CodeGen.

Model	**N**	**Under-stand**	**Agree**	**Use**
		Dataset: CoNaLa		
CodeGen *FT:C*	58	0.5345	0.3966	0.4310
CodeGen *FT:SO*	68	0.0882	-0.1471	-0.1912
CodeGen *FT:C+*	56	0.8571	0.4464	**0.4286**
GPT-Neo *FT:C*	62	0.0000	-0.4677	-0.5323
GPT-Neo *FT:SO*	55	-0.0364	-0.6364	-0.8364
CoPilot	39	**0.8974**	**0.4872**	0.2308
		Dataset: SO		
CodeGen *FT:C*	25	**0.9200**	**-0.3200**	**-0.2000**
CodeGen *FT:SO*	13	0.0769	-0.6154	-0.4615
GPT-Neo	21	-0.9048	-1.8095	-1.8095
GPT-Neo *FT:C*	21	0.2381	-0.5238	-0.4286
GPT-Neo *FT:SO*	16	-0.6250	-1.1875	-1.1250

Table 2. Human perceiving evaluation

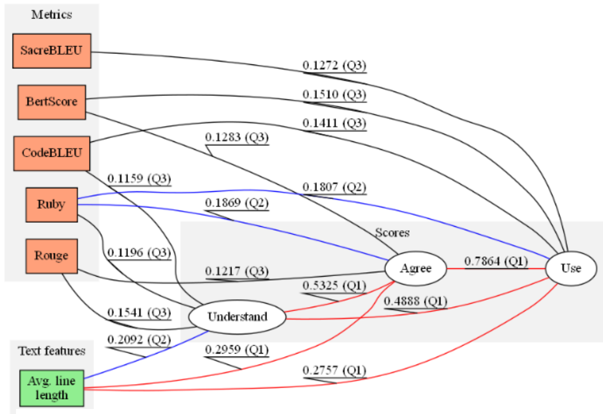

Figure 2. Mutual information and feature connection

For the analysis of MI, we divided the selection of the connections between features (FG2-FG2, FG2-FG1, FG2-FG3) into four groups according to the quartiles of the MI distribution (the division levels correspond to MI of 0.1144, 0.1621, 0.2683 for thresholds between Q4-Q3, Q3-Q2, Q2-Q1 respectively). We dropped Q4 (lowest MI) connections as insignificant and considered the others for further analysis. Figure 2 shows the selection of features connected with the labels denoting MI level and the quartile to which the value belongs. Also, the quartiles are shown in color (Q1 – red, Q2 – blue, Q3 – black).

As expected, the internal connections of perceiving features have high MI. The highest interconnection is observed between Agree and Use scores. Additionally, they are highly correlated ($cor = 0.8981$). A simple linear regression model was estimated as $U = 0.8389A + 0.0764C + 0.0134$ ($R^2 = 0.8098$) where U is for intention to use, A is for agreement, C is for understanding (internal consistency). Thus, we expect that the intention to use is highly defined by the agreement to the answer. On the other hand, the dependency between understanding and agreement can also be observed, but is rather lower: $A = 0.6687C + 0.0376$ (with $R^2 = 0.4358$).

We see that the connection of most of the NLP metrics is rather low (Q3), except for the Ruby metric showing a more significant (Q2) influence on the agreement and the intention to use. This can be interpreted as a good evaluation of code quality. The remaining features are mostly low and have interconnection with different perceiving scores. In general, the MI for interconnection with the intention to use is slightly higher.

The analysis shows that basic linguistic features (FG3) mainly have low interconnection with perceiving votes (FG2) as the connections were assessed with low MI (Q4) with one exception for average line length having high (Q1 or Q2) impact on the perceiving features. Further analysis shows that there are weak results where a model failed to generate a structured answer and produce long (mainly erroneous) lines of code.

6 Discussion

The study focused on code generation problems. For this purpose, we used modern NLG models (CodeGen, GPT) and performed fine-tuning to obtain a higher quality of the results according to the common pipeline. The results we obtained with the CodeGen model look promising and produce high-quality results comparable to the industrial solutions (CoPilot). Still, we consider the further improvement of the code generation QA model as one of our future research directions.

The main goal of the study is the investigation of the human perceiving structure in the NLG problem. The area of human evaluation is widely studied in different NLG settings including general assessment of natural language generation, as well as distinguishing model-generated from human-generated answer. The common goal of most studies in the area is model improvement with collected evaluation feedback. Still, the question of how we can evaluate human feedback and which level of trust can be given to it is still open. This problem becomes more challenging if the evaluation is performed in a complex domain where a human annotator needs to be a domain expert. In that case, the human feedback collection could be more expensive and can provide more diversity in judgments. With this in mind, we considered a problem of NLG in programming QA and code generation, with programmers as experts evaluating high-level correspondence of the answer to a proposed question. Within the study, we focused on the general structure of human perceiving divided into three main parts: whether the human understands the model output; whether he/she considers it as correct; whether he/she is ready to use it in practice. This structure of human perceiving was considered previously in the expert-based evaluation of decision support systems (Kovalchuk et al., 2022) and showed good results in understanding human intentions in perceiving AI-based prediction in such complex domains as medicine.

Within the study, we analyzed internal interconnection between human perceiving features in code generation and discovered that although the agreement and intention to use are highly correlated, the understanding (subjective correctness) and agreement show lower interconnections. This could be interpreted as a sign of existing issues in generated code properly recognized by a human, i.e., the answer generated by the model “looks like code” but doesn’t resolve the question properly. On the other hand, the high interconnection between agreement and intention to use could be treated as promising results for code generation problems: if the code answers the question, it is good enough to be used.

An interesting result obtained in the analysis of external interconnection of perceiving score is a rather weak MI of connections to the common NLP metrics. The only metric showing medium interconnection is Ruby, intentionally developed for code evaluation with semantic comparison (Tran et al., 2019). We consider this result as a sign of the rather weak applicability of common NLP metrics to complex NLG problems.

One of the questions raised during the evaluation is whether we can compare the syntactic correctness of the code to the subjective correctness (understanding) of the code. During the evaluation, we see several examples of code that was syntactically incorrect but may have been evaluated as useful (e.g. containing the correct line of code followed by an ill-formatted line). We believe that continuous Likert-scale-based evaluation may help to improve the model training/finetuning in further studies with human evaluation. Still, we consider this issue as one of the directions for the future development of the proposed approach.

It is worth mentioning that CodeGen FT:SO shows lower performance compared to CodeGen FT:C even on the SO dataset. We suppose that a possible reason for such behavior is the fact that the CoNaLa dataset was manually curated and contains only one-line code answers. At the same time, the SO dataset was not filtered in that way and contains answers of diverse length, structure, and quality. A further investigation of this issue is one of the directions for further research.

The presented results can be used in two ways. First, to improve models by training with a deeper understanding of human perceiving structure. For example, the mentioned values could be considered as a sequential filter where the next step can be considered only by the answer passed from the previous one. In this way, the perceiving may be considered as a reward for the model in the generation of the answer.

Second, the understanding of human perceiving may improve human-computer interaction in AI-based applications. For instance, the separate prediction of human perceiving features may provide important information depending on the interaction scenario. In particular, the Agree score may be more influential in code automatically generated by explicit specification, while the Use score may be more important in direct human-centered QA (e.g., in a form of an intelligent IDE assistant).

7 Conclusion and future work

The current study shows early results in the research of human perceiving understanding within the context of NLG human evaluation. Being aimed at a deeper understanding of the internal structure of human perceiving and interconnection with common metrics, the study shows that perceiving structure may be decomposed into a complex value with implicit interconnection directing from model-generated structure evaluation to subjective intention to use. The proposed structure of human perceiving may be further used for collecting judgments in complex domains where direct application of common NLP metrics gives rather weak results and where a high semantic diversity in the possible answer may be observed.

We consider the further development of the proposed study in the following directions. First, we would like to continue collecting human feedback in order to advance the development of the perceiving model. Additionally, we would like to extend the research to analyze the personal characteristics of the human (in our case, it may be a personal experience, relevance to the question, etc.). Second, we would like to investigate the possible application of the decomposed human perceiving value to model training/finetuning. Third, we are interested in the extension of the model with technical characteristics of generated code (e.g. checking for syntactic errors, test-based evaluation). Finally, we are planning to investigate the influence of the context on human perceiving including the dependencies from the application scenario.

Acknowledgements

We thank all the users involved into the evaluation procedure presented in this study for their effort in assessment of the generated answers. We are also grateful to the anonymous GEM reviewers for their valuable comments and suggestions.

References

Ajzen, I. (1991). The theory of planned behavior. *Organizational Behavior and Human Decision Processes*, *50*(2), 179–211. https://doi.org/10.1016/0749-5978(91)90020-T

Beyer, S., Macho, C., Di Penta, M., & Pinzger, M. (2020). What kind of questions do developers ask on Stack Overflow? A comparison of automated approaches to classify posts into question categories. *Empirical Software Engineering*, *25*(3), 2258–2301. https://doi.org/10.1007/s10664-019-09758-x

Black, Sid, Leo, Gao, Wang, Phil, Leahy, Connor, & Biderman, Stella. (2021). *GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow* (1.0). Zenodo. https://doi.org/10.5281/ZENODO.5297715

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. de O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., Ray, A., Puri, R., Krueger, G., Petrov, M., Khlaaf, H., Sastry, G., Mishkin, P., Chan, B., Gray, S., … Zaremba, W. (2021). *Evaluating Large Language Models Trained on Code* (arXiv:2107.03374). arXiv. http://arxiv.org/abs/2107.03374

De Mattei, L., Lai, H., Dell'Orletta, F., & Nissim, M. (2021). Human Perception in Natural Language Generation. *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021)*, 15–23. https://doi.org/10.18653/v1/2021.gem-1.2

Evtikhiev, M., Bogomolov, E., Sokolov, Y., & Bryksin, T. (2022). *Out of the BLEU: How should we assess quality of the Code Generation models?* (arXiv:2208.03133). arXiv. http://arxiv.org/abs/2208.03133

Hämäläinen, M., & Alnajjar, K. (2021). Human Evaluation of Creative NLG Systems: An Interdisciplinary Survey on Recent Papers. *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics*

(GEM 2021), 84–95. https://doi.org/10.18653/v1/2021.gem-1.9

Kovalchuk, S. V., Kopanitsa, G. D., Derevitskii, I. V., Matveev, G. A., & Savitskaya, D. A. (2022). Three-stage intelligent support of clinical decision making for higher trust, validity, and explainability. *Journal of Biomedical Informatics*, *127*, 104013. https://doi.org/10.1016/j.jbi.2022.104013

Lu, S., Guo, D., Ren, S., Huang, J., Svyatkovskiy, A., Blanco, A., Clement, C., Drain, D., Jiang, D., Tang, D., Li, G., Zhou, L., Shou, L., Zhou, L., Tufano, M., Gong, M., Zhou, M., Duan, N., Sundaresan, N., … Liu, S. (2021). *CodeXGLUE: A Machine Learning Benchmark Dataset for Code Understanding and Generation* (arXiv:2102.04664). arXiv. http://arxiv.org/abs/2102.04664

Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., Hesse, C., Jain, S., Kosaraju, V., Saunders, W., Jiang, X., Cobbe, K., Eloundou, T., Krueger, G., Button, K., Knight, M., Chess, B., & Schulman, J. (2022). *WebGPT: Browser-assisted question-answering with human feedback* (arXiv:2112.09332). arXiv. http://arxiv.org/abs/2112.09332

Nijkamp, E., Pang, B., Hayashi, H., Tu, L., Wang, H., Zhou, Y., Savarese, S., & Xiong, C. (2022). *A Conversational Paradigm for Program Synthesis* (arXiv:2203.13474). arXiv. http://arxiv.org/abs/2203.13474

Ren, S., Guo, D., Lu, S., Zhou, L., Liu, S., Tang, D., Sundaresan, N., Zhou, M., Blanco, A., & Ma, S. (2020). *CodeBLEU: A Method for Automatic Evaluation of Code Synthesis* (arXiv:2009.10297). arXiv. http://arxiv.org/abs/2009.10297

Stiennon, N., Ouyang, L., Wu, J., Ziegler, D., Lowe, R., Voss, C., Radford, A., Amodei, D., & Christiano, P. F. (2020). Learning to summarize with human feedback. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, & H. Lin (Eds.), *Advances in Neural Information Processing Systems* (Vol. 33, pp. 3008–3021). Curran Associates, Inc. https://proceedings.neurips.cc/paper/2020/file/1f89885d556929e98d3ef9b86448f951-Paper.pdf

Tran, N., Tran, H., Nguyen, S., Nguyen, H., & Nguyen, T. (2019). Does BLEU Score Work for Code Migration? *2019 IEEE/ACM 27th International Conference on Program Comprehension (ICPC)*, 165–176. https://doi.org/10.1109/ICPC.2019.00034

Xu, F. F., Alon, U., Neubig, G., & Hellendoorn, V. J. (2022). *A Systematic Evaluation of Large Language Models of Code* (arXiv:2202.13169). arXiv. http://arxiv.org/abs/2202.13169

Yin, P., Deng, B., Chen, E., Vasilescu, B., & Neubig, G. (2018). Learning to mine aligned code and natural language pairs from stack overflow. *Proceedings of the 15th International Conference on Mining Software Repositories*, 476–486. https://doi.org/10.1145/3196398.3196408

Zhong, M., Liu, G., Li, H., Kuang, J., Zeng, J., & Wang, M. (2022). *CodeGen-Test: An Automatic Code Generation Model Integrating Program Test Information* (arXiv:2202.07612). arXiv. http://arxiv.org/abs/2202.07612