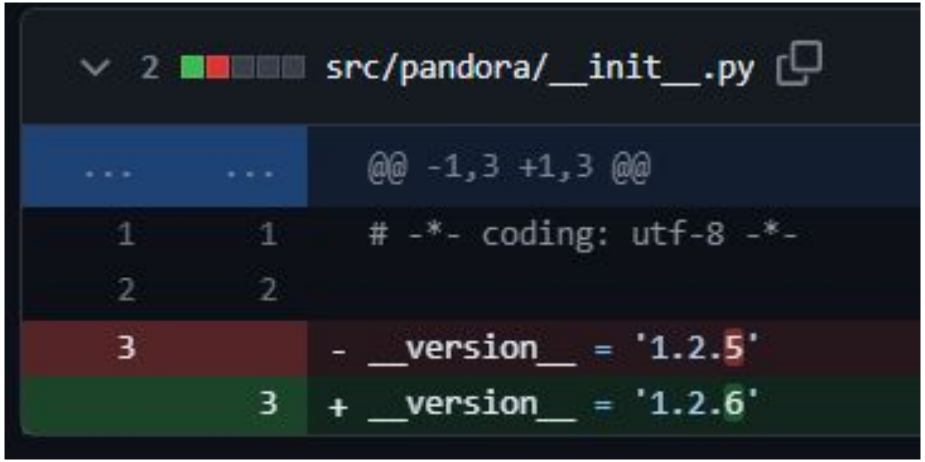
Generation of commit messages with deep learning methods

Sergeev Nikita BS20-AI
Supervisor: Vladimir Ivanov

Problem statement

Commit - snapshot of changes in code.
Commit message - natural language text to describe the changes in the code

Goal: research the methods to automatically generate commit messages



```
2 src/pandora/__init__.py
@@ -1,3 +1,3 @@
1 1  # -*- coding: utf-8 -*-
2 2
3   - __version__ = '1.2.5'
  3 + __version__ = '1.2.6'
```

Commit message: *change version*

Literature review

Model architectures

- Transformer based models
- Transformer + GNN (AST)
- Transformer + Retrieval

Dataset	**diffs**	**samples**	**metadata**	**PLs**
CommitBERT	+	345K	-	6
MCMD	+	450K	+	5
Commit-Chronicle	+	10.7M	+	20
Parsed	+	300K	+	1

Literature review

Evaluation metrics

- BLEU (normalized version)
- Exact match
- Edit similarity
- BERTScore (was not used in the literature)

Research gaps

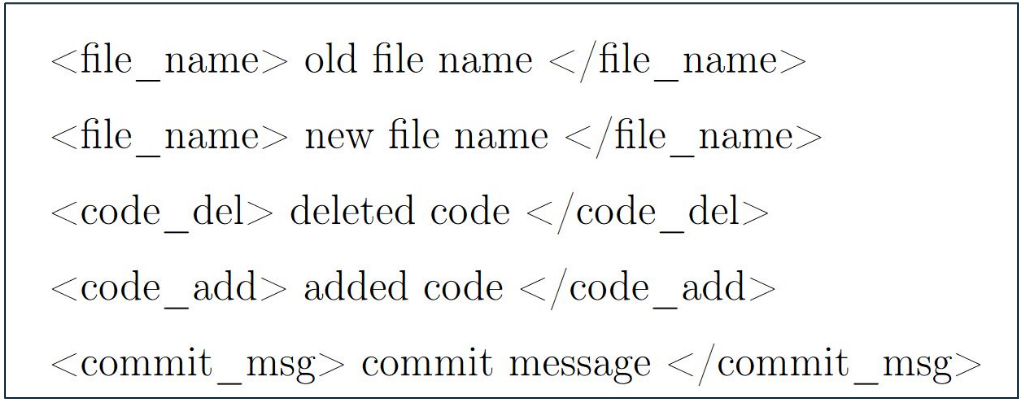
- Inability to handle big commits
- Lack of semantic evaluation metrics
- Adaptation to the repository style of commit messages
- Lack of experiments with LLMs

Addressed problems

	Problem	**Solution**
1	Lack of experiments with LLMs	Scaling the base model
2	Handling big commits	File attention model
3	Style adaptation	Model with the re
4	Lack of evaluation metrics	BERTScore metric applied

Baseline model

- CodeT5+ - transformer model with 220 Million parameters
- Used as a base model for all further experiments
- Added special tokens for better capturing of the semantic

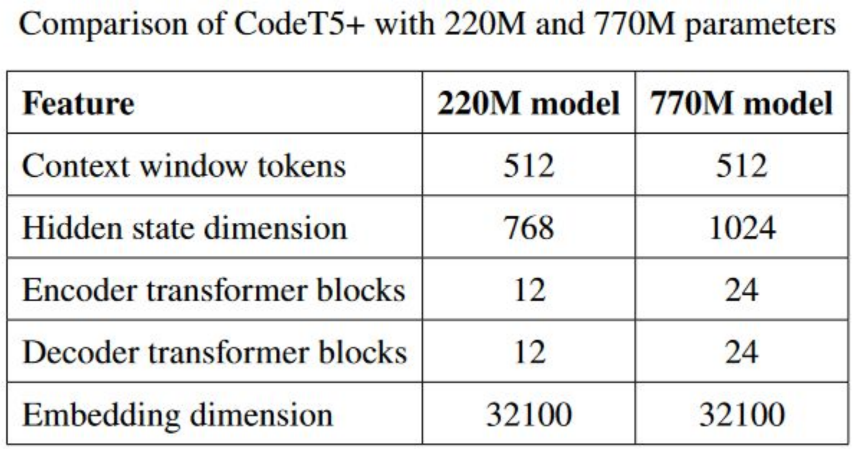


```
<file_name> old file name </file_name>
<file_name> new file name </file_name>
<code_del> deleted code </code_del>
<code_add> added code </code_add>
<commit_msg> commit message </commit_msg>
```

Input format for the model

Larger version of the baseline

Scaled version is not able to process bigger commits, but have better ability to capture the semantic of code



Comparison of CodeT5+ with 220M and 770M parameters

Feature	**220M model**	**770M model**
Context window tokens	512	512
Hidden state dimension	768	1024
Encoder transformer blocks	12	24
Decoder transformer blocks	12	24
Embedding dimension	32100	32100

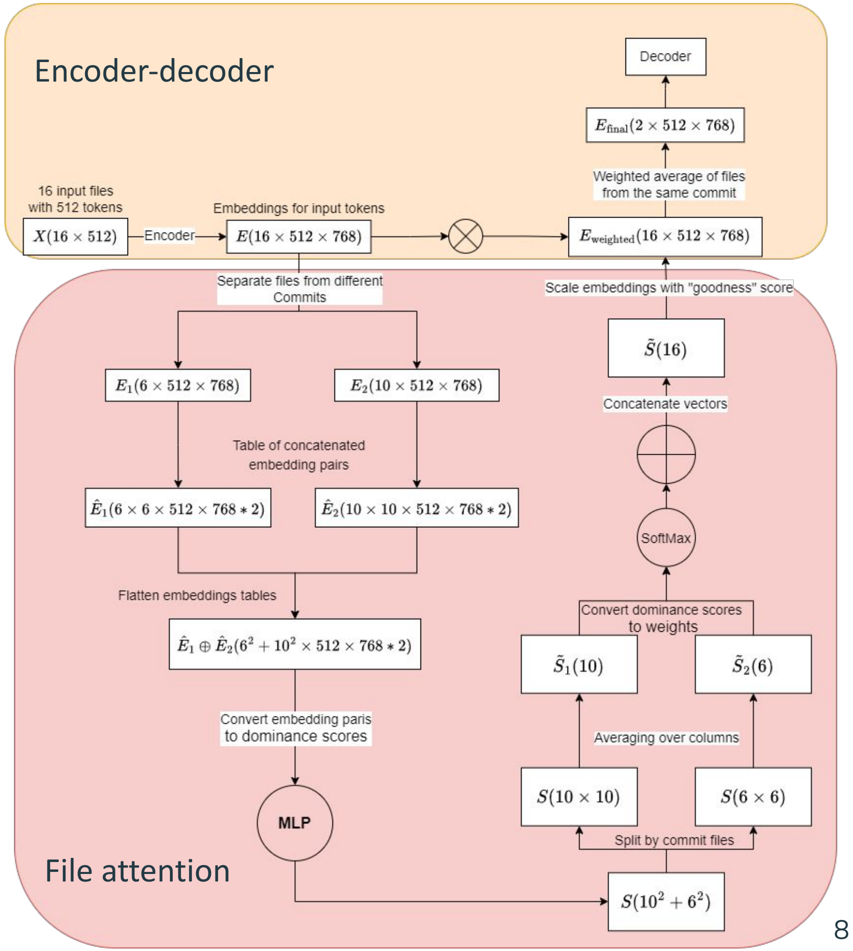
Big commits handling

Problem:

- Used transformer models have limited context
- Big commits is typically consists of multiple files

Solution:

- To effectively process multiple file - handle embedding of each file separately
- Before passing embeddings to the decoder - weighted average them with trainable coefficients



Encoder-decoder
Decoder
$E_{\text{final}}(2 \times 512 \times 768)$
Weighted average of files from the same commit
16 input files with 512 tokens
Embeddings for input tokens
$X(16 \times 512)$
Encoder
$E(16 \times 512 \times 768)$
$E_{\text{weighted}}(16 \times 512 \times 768)$
Separate files from different Commits
Scale embeddings with "goodness" score
$\tilde{S}(16)$
$E_1(6 \times 512 \times 768)$
$E_2(10 \times 512 \times 768)$
Concatenate vectors
Table of concatenated embedding pairs
$\hat{E}_1(6 \times 6 \times 512 \times 768 * 2)$
$\hat{E}_2(10 \times 10 \times 512 \times 768 * 2)$
SoftMax
Flatten embeddings tables
Convert dominance scores to weights
$\hat{E}_1 \oplus \hat{E}_2(6^2 + 10^2 \times 512 \times 768 * 2)$
$\tilde{S}_1(10)$
$\tilde{S}_2(6)$
Convert embedding paris to dominance scores
Averaging over columns
$S(10 \times 10)$
$S(6 \times 6)$
MLP
Split by commit files
File attention
$S(10^2 + 6^2)$

Style adaptation

- To adapt the message to the repository style - add retrieval module to the base model
- Input is augmented with two additional messages:
  1) Previous message from the same repository
  2) Message from the semantically closest commit from database

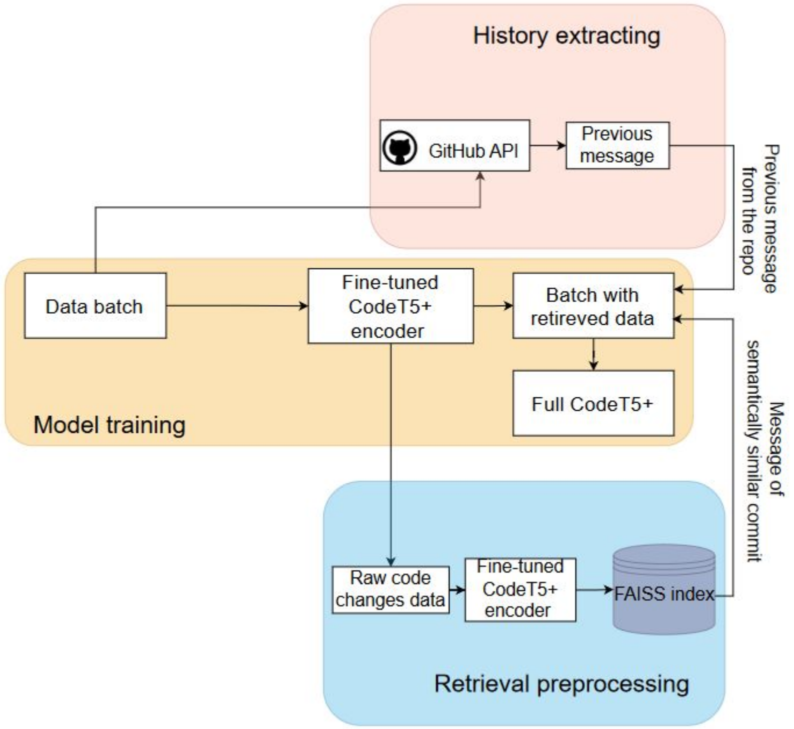
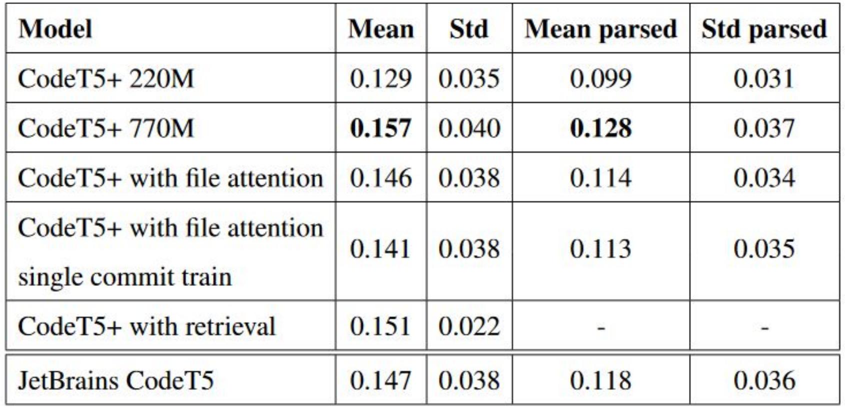

History extracting
GitHub API
Previous message
Previous message from the repo
Data batch
Fine-tuned CodeT5+ encoder
Batch with retireved data
Full CodeT5+
Model training
Message of semantically similar commit
Raw code changes data
Fine-tuned CodeT5+ encoder
FAISS index
Retrieval preprocessing

Fig. 4.5. Architecture description for CodeT5+ with retrieval.

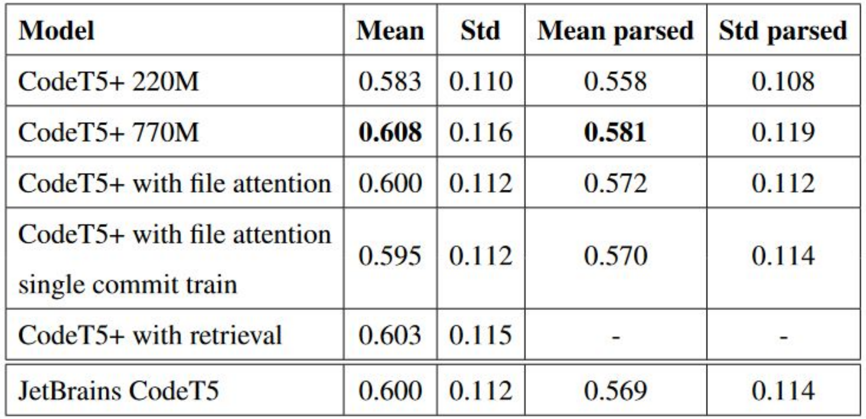
Quality of generation

Normalized BLEU score



Model	Mean	Std	Mean parsed	Std parsed
CodeT5+ 220M	0.129	0.035	0.099	0.031
CodeT5+ 770M	**0.157**	0.040	**0.128**	0.037
CodeT5+ with file attention	0.146	0.038	0.114	0.034
CodeT5+ with file attention single commit train	0.141	0.038	0.113	0.035
CodeT5+ with retrieval	0.151	0.022	-	-
JetBrains CodeT5	0.147	0.038	0.118	0.036

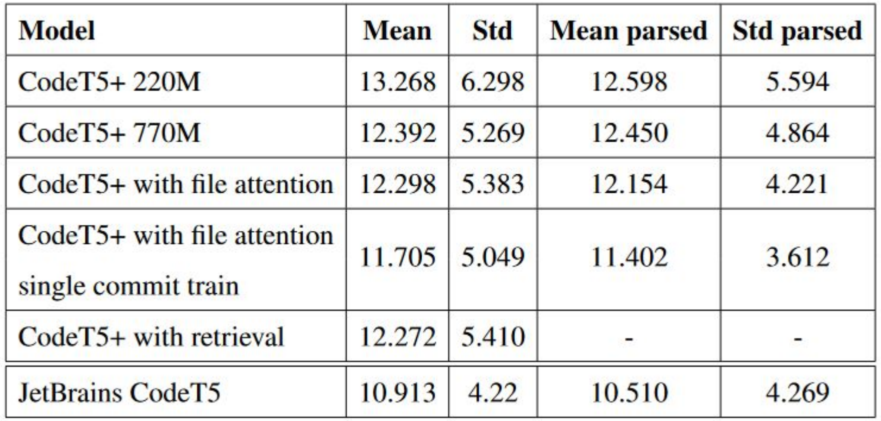
BERTScore



Model	Mean	Std	Mean parsed	Std parsed
CodeT5+ 220M	0.583	0.110	0.558	0.108
CodeT5+ 770M	**0.608**	0.116	**0.581**	0.119
CodeT5+ with file attention	0.600	0.112	0.572	0.112
CodeT5+ with file attention single commit train	0.595	0.112	0.570	0.114
CodeT5+ with retrieval	0.603	0.115	-	-
JetBrains CodeT5	0.600	0.112	0.569	0.114

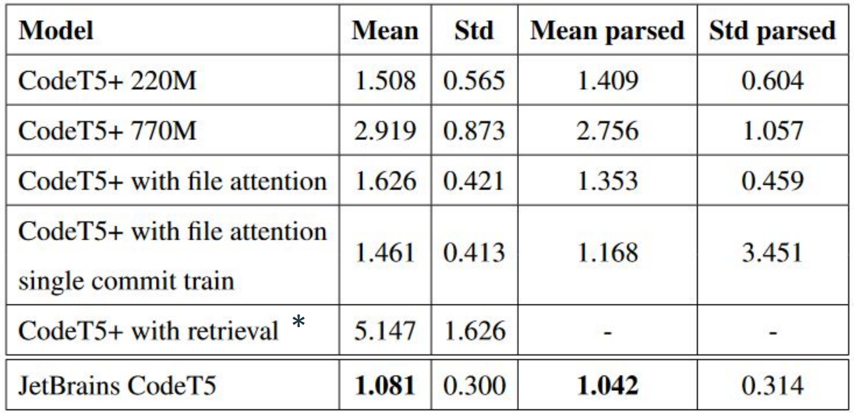
Models performance

Message length



Model	Mean	Std	Mean parsed	Std parsed
CodeT5+ 220M	13.268	6.298	12.598	5.594
CodeT5+ 770M	12.392	5.269	12.450	4.864
CodeT5+ with file attention	12.298	5.383	12.154	4.221
CodeT5+ with file attention single commit train	11.705	5.049	11.402	3.612
CodeT5+ with retrieval	12.272	5.410	-	-
JetBrains CodeT5	10.913	4.22	10.510	4.269

Evaluation time



Model	Mean	Std	Mean parsed	Std parsed
CodeT5+ 220M	1.508	0.565	1.409	0.604
CodeT5+ 770M	2.919	0.873	2.756	1.057
CodeT5+ with file attention	1.626	0.421	1.353	0.459
CodeT5+ with file attention single commit train	1.461	0.413	1.168	3.451
CodeT5+ with retrieval *	5.147	1.626	-	-
JetBrains CodeT5	**1.081**	0.300	**1.042**	0.314

** Batch size used for retrieval model - 64.*
And for other models - 20.

Conclusion

1. Special tokens lead to the performance degradation of the model
2. File attention model did not boost the performance of the model
3. Scaling the model increased the model performance, while preserving acceptable efficiency
4. Retrieval module make the BLEU metric more robust and increased the performance
5. BERTScore correlates with the quality of messages

Future work

1) Remove special tokens
2) More deeply analyse the work of the file attention module
3) Combine file attention with the retrieval model

Thank you!

BERTScore analysis

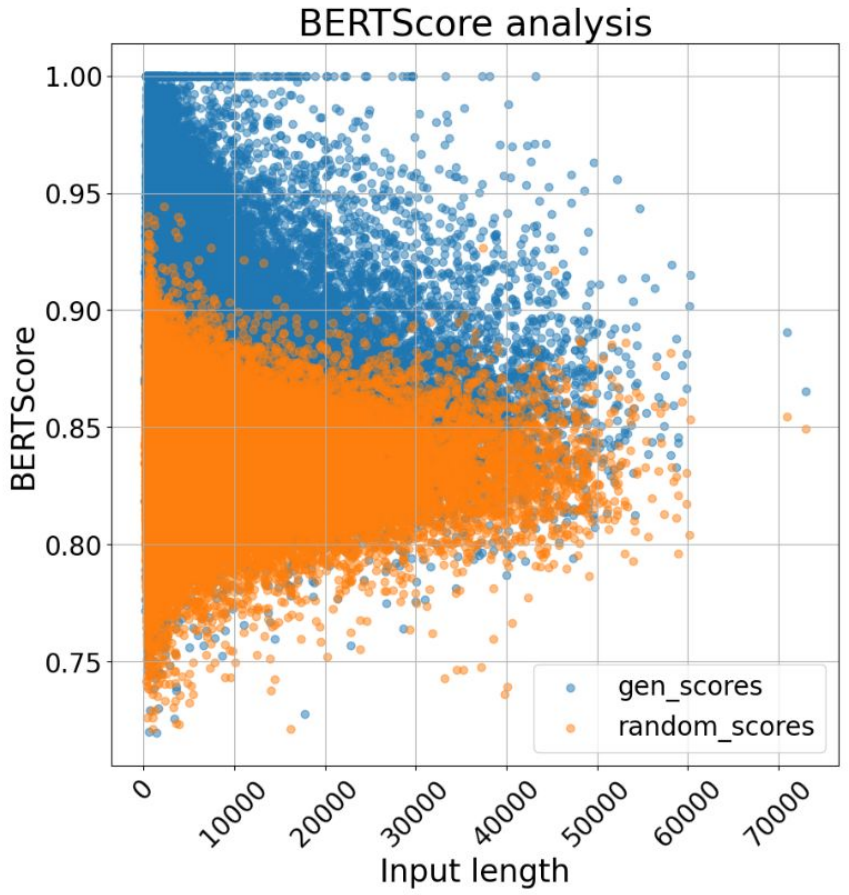
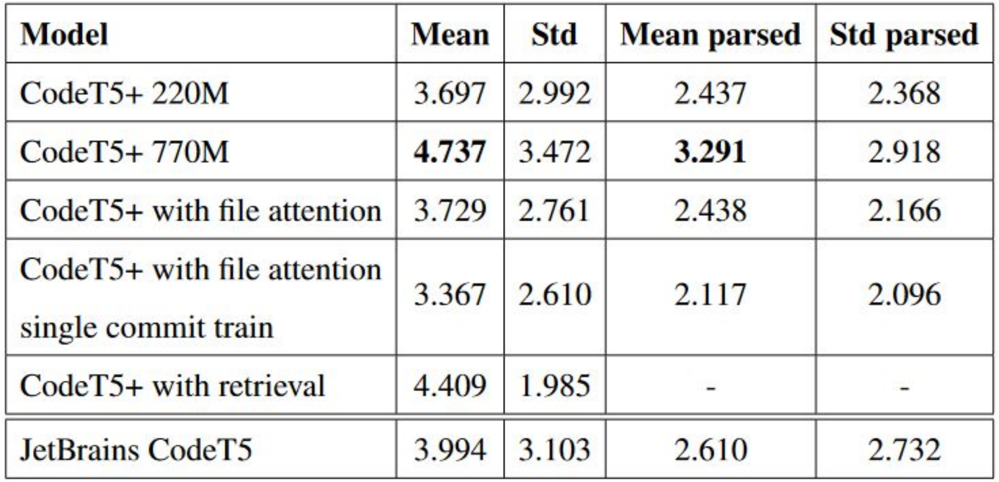


Fig. A.1. Comparing the BERTScore of generated messages and random ones.

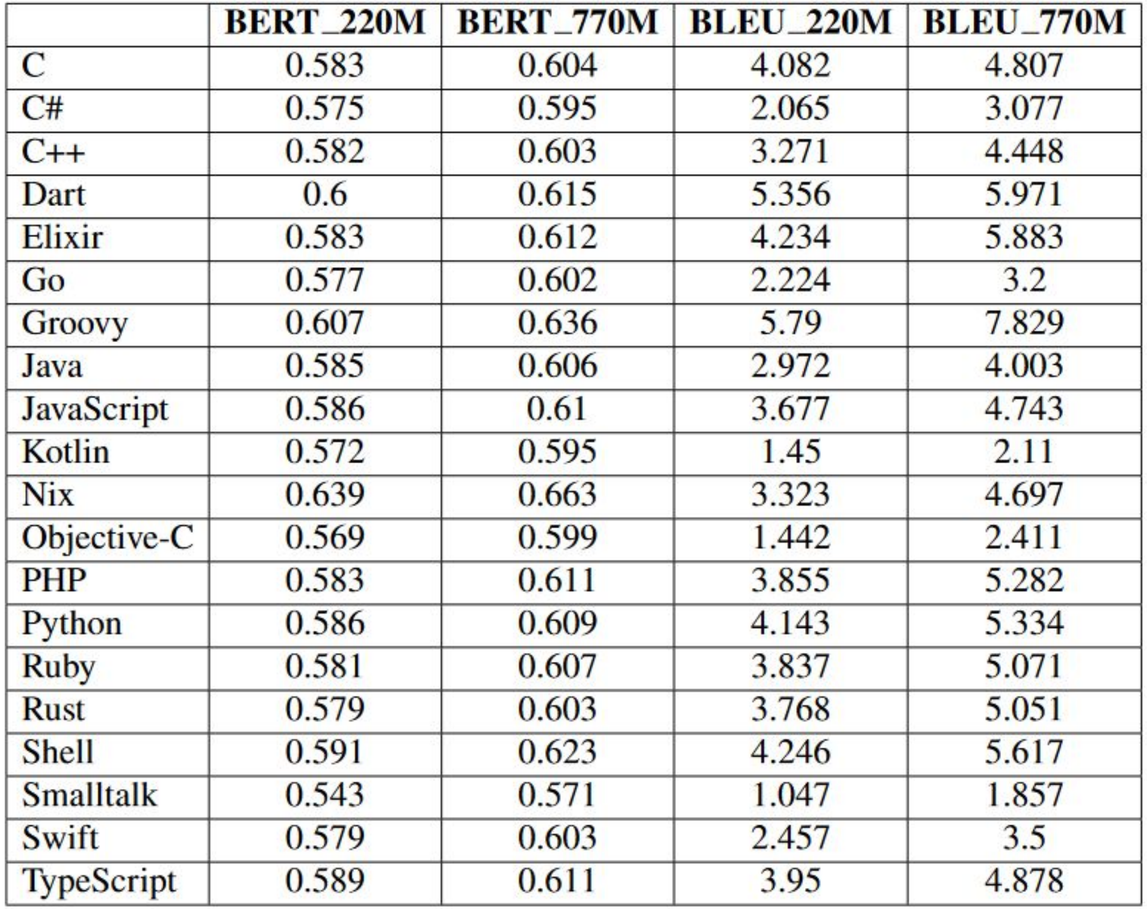
BLEU score results

BLEU scores



Model	Mean	Std	Mean parsed	Std parsed
CodeT5+ 220M	3.697	2.992	2.437	2.368
CodeT5+ 770M	**4.737**	3.472	**3.291**	2.918
CodeT5+ with file attention	3.729	2.761	2.438	2.166
CodeT5+ with file attention single commit train	3.367	2.610	2.117	2.096
CodeT5+ with retrieval	4.409	1.985	-	-
JetBrains CodeT5	3.994	3.103	2.610	2.732

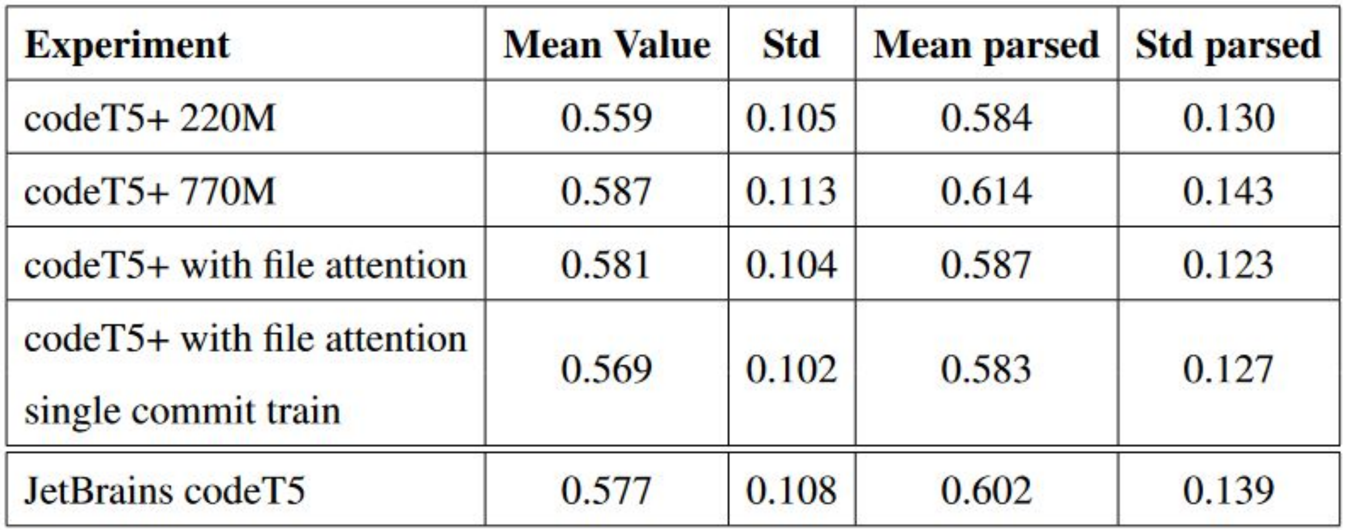
Performance for the specific programming language



	BERT_220M	BERT_770M	BLEU_220M	BLEU_770M
C	0.583	0.604	4.082	4.807
C#	0.575	0.595	2.065	3.077
C++	0.582	0.603	3.271	4.448
Dart	0.6	0.615	5.356	5.971
Elixir	0.583	0.612	4.234	5.883
Go	0.577	0.602	2.224	3.2
Groovy	0.607	0.636	5.79	7.829
Java	0.585	0.606	2.972	4.003
JavaScript	0.586	0.61	3.677	4.743
Kotlin	0.572	0.595	1.45	2.11
Nix	0.639	0.663	3.323	4.697
Objective-C	0.569	0.599	1.442	2.411
PHP	0.583	0.611	3.855	5.282
Python	0.586	0.609	4.143	5.334
Ruby	0.581	0.607	3.837	5.071
Rust	0.579	0.603	3.768	5.051
Shell	0.591	0.623	4.246	5.617
Smalltalk	0.543	0.571	1.047	1.857
Swift	0.579	0.603	2.457	3.5
TypeScript	0.589	0.611	3.95	4.878

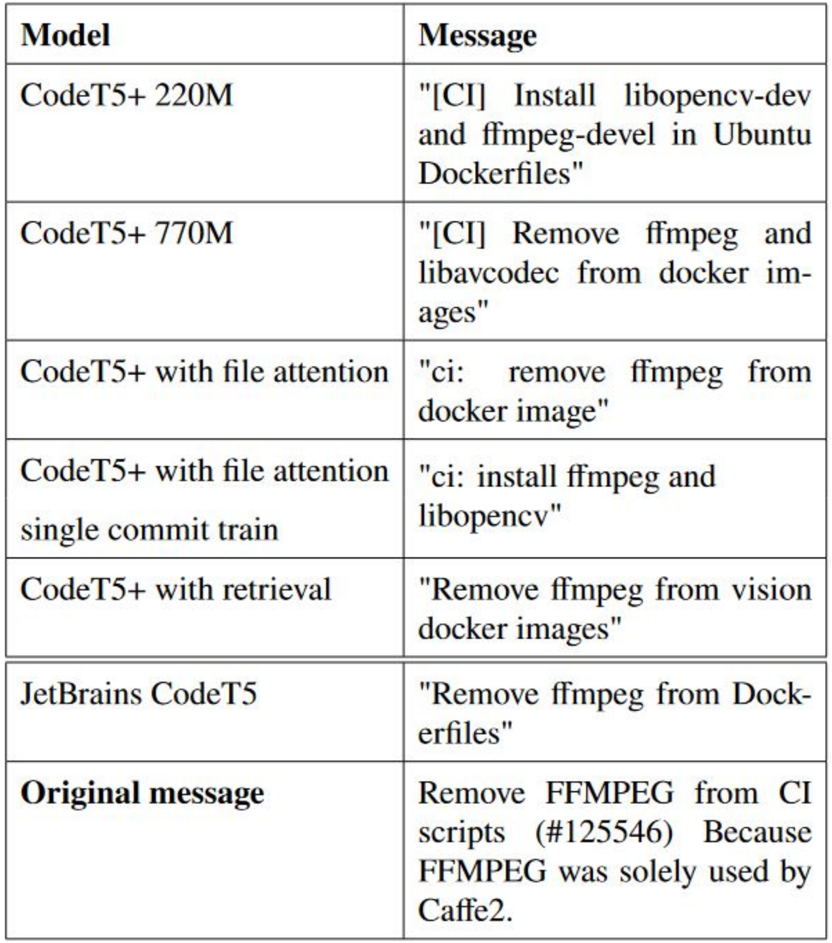
Results for commits with >13 files

BERTScore for big commits



Experiment	Mean Value	Std	Mean parsed	Std parsed
codeT5+ 220M	0.559	0.105	0.584	0.130
codeT5+ 770M	0.587	0.113	0.614	0.143
codeT5+ with file attention	0.581	0.104	0.587	0.123
codeT5+ with file attention single commit train	0.569	0.102	0.583	0.127
JetBrains codeT5	0.577	0.108	0.602	0.139

Inference example



Model	**Message**
CodeT5+ 220M	"[CI] Install libopencv-dev and ffmpeg-devel in Ubuntu Dockerfiles"
CodeT5+ 770M	"[CI] Remove ffmpeg and libavcodec from docker images"
CodeT5+ with file attention	"ci: remove ffmpeg from docker image"
CodeT5+ with file attention single commit train	"ci: install ffmpeg and libopencv"
CodeT5+ with retrieval	"Remove ffmpeg from vision docker images"
JetBrains CodeT5	"Remove ffmpeg from Dockerfiles"
Original message	Remove FFMPEG from CI scripts (#125546) Because FFMPEG was solely used by Caffe2.