A Dataset for Programming-based Instructional Video Classification and Question Answering

Sana Javaid Raja, Adeel Zafar, Aqsa Shoaib
Faculty of Computing, Riphah International University
Correspondence: sanajavaidraja@gmail.com, adeel.zafar@riphah.edu.pk

Abstract

This work aims to develop an understanding of the rapidly emerging field of VideoQA, particularly in the context of instructional programming videos. It also encourages the designing of a system that can produce visual answers to programming-based natural language questions. We introduce two datasets: CodeVidQA, with 2,104 question-answer pair with timestamps and links taken from programming videos extracted using Stack Overflow for Programming Visual Answer Localization task, and CodeVidCL with 4,291 videos (1751 programming, 2540 non-programming) for Programming Video Classification task. In addition, we proposed a framework that adapts BigBird and SVM for video classification techniques. The proposed approach achieves a significantly high accuracy of 99.61% for video classification.

1 Introduction

One of the most interesting trends on the internet is the availability of information in the form of videos. Similarly, instructional videos have become the usual way of teaching and learning how to solve certain problems. Programming-based instructional videos have emerged and effectively convey targeted information via instructional demonstrations and voice-overs. 80% of all of those videos are in English, 2% are in Spanish and 4% are in German (Kadriu et al., 2020).

Hence, the practice of self-learning has become more convenient, especially with the availability of massive open online courses (MOOCs) (Hill, 2014). Approximately 19% to 20% of a developer's time is dedicated to using the internet for purposes of searching information for development and programming. Although more programming tutorials are being produced and used, developers are sometimes hard-pressed to find good videos that give as much coverage to actual use cases as they do to the theory behind them. For example, developers looking for specific questions may scan through many videos to find one that they want. Moreover, during tutorial sessions, language barriers may impede the learner's comprehension of the subject, especially among those whose second language is English (Brandt et al., 2009).

Despite being widely believed as popular, Video Question Answering does not escape from inherently challenging hurdles. To answer questions, VideoQA models need adequate knowledge of the visual content so they can recognize visual objects and understand their semantic, spatial, temporal, and causal relations (Khurana and Deshpande, 2021). However, information on their effectiveness remains limited, often certainly, because of the absence of proper frameworks and datasets for evaluation. Uncovering the causes behind failure in VideoQA is a tough nut to crack, either posed by the dataset or the trained model (Khurana and Deshpande, 2021).

The contributions of this work are the new CodeVidQA dataset which contains 2,104 comprehensively annotated timestamped question-answer pairs curated from videos extracted from Stack Overflow and a new dataset, CodeVidCL consists of 4,291 programming and other related videos. We also suggest that the employment of BigBird and SVM models in the structure of the ensemble would increase the efficiency of VideoQA and the classification of programming videos.

The CodeVidQA dataset facilitates advancements in VideoQA systems by offering timestamped question-answers pairs for precise video-based educational content retrieval. Similarly, the CodeVidCL dataset supports the classification and analysis of programming tutorial videos, enhancing AI-driven educational applications. Both datasets are publicly available on GitHub, promoting accessibility and further research in the domain[1].

2 Literature survey

2.1 Existing Datasets for VideoQA

One of the first novel datasets used for VideoQA is the YouTube2Text(Guadarrama et al., 2013) dataset, including 1,987 videos and 122,708 natural language descriptions. Other similar datasets include Movie QA (Lei et al., 2019), VideoQA (Mun et al., 2017), and TVQA+ (Kim et al., 2017) covering different types of videos.

Further, MarioQA (Calzolari et al., 2020) is based on a game, Pororo-QA (Gupta et al., 2023) is based on cartoon and life scenarios like LifeQA (Lei et al., 2018) and MedVidQA (Hamon et al., 2017).

2.2 Video Question Answering in Natural Language Processing

VideoQA is an extended problem in Natural Language Processing NLP in which question answering is performed through the contents of the video sequence. It is also divided into multi-choice QA, where models choose from the options available choices, and open-ended QA which involves generation, regression, or classification (Choi et al., 2021).

The main problem of VQA is manifested in the need to accurately identify the correct answers based on the comprehension of the context of the video. For example, the proposed models process keyframes by using attention mechanisms, or apply knowledge-guided methods for further complex queries. In particular, a Siamese Sampling and Reasoning (SiaSamRea) has been proven to achieve initial success across multiple benchmarks, improving the performance on MSRVTT-QA(Xu et al., 2017), MSVD-QA(Xu et al., 2017) and Activity Net-QA(Yu et al., 2019) datasets(Yu et al., 2024).

2.3 Need for Programming-based Instructional Video QA Dataset

Simply predicting natural language answers to most of the questions don't reflect real-world interaction as people want to follow visuals step by step along with textual answers. Therefore, recent developments in video question-answering systems on specific domains like medicine, movies and games, etc. there is need to design systems that are related to programming. Although there are datasets for entertainment, such as MovieQA (Lei et al., 2019), or TVQA (Tapaswi et al., 2016), there is a lack of datasets specifically on programming instruction.

2.4 Need for Programming-Based Instructional Video Classification Dataset

A large amount of programming instructional videos are used in the learning process to develop programming skills, but there is a lack of efficient methods for classifying them that need domain-specific datasets for the Programming Video Classification task. This paper establishes that there are several methods of classifying videos through the use of video transcripts and contextual features. Specifically, Kinetics (Lopez et al., 2007) which is related to human actions and COIN (Gupta et al., 2023) which consists of 11,827 instructional videos collect from 12 domains.

3 Material and Methods

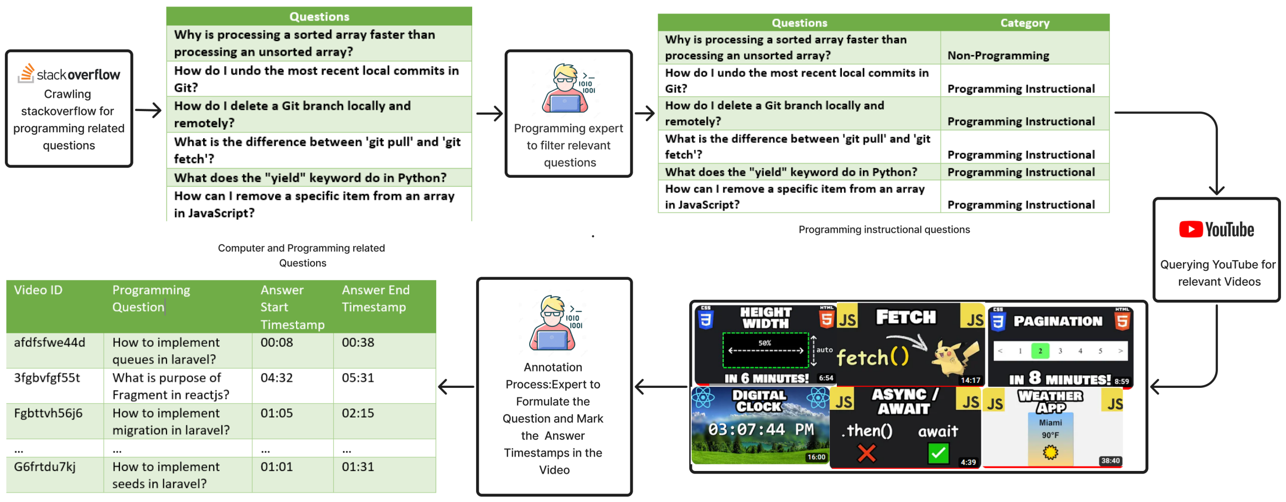
3.1 CodeVidQA Creation

The selection of several videos is required to construct a high-quality programming instructional VideoQA dataset from several general programming languages and databases such as Java, Python, JavaScript, MySQL, Oracle, and MongoDB, can be selected.

Real-life questions, such as, "How can I use queues in Laravel?" Counterarguments to the above arguments can only be effected by practical implementation, as theoretical answers to the problems can hardly be comprehended. The dataset creation process is initiated by systematically pulling questions from the Stack Overflow website, where specific questions related to programming and non-programming instructional were identified. After that annotation was performed on programming instructional videos extracted from YouTube that results in the generation of question brackets together with timestamped answers by two programming experts.

Both programming experts have more than two years of experience in developing programs and possess proficient knowledge of more than one programming language. For annotation purposes, programming instructional questions were divided equally between both experts for the formulation of the resulting dataset. The following schematic outlines the key steps involved in the methodology for the CodeVidQA dataset. Each component illustrates the processes and relationships integral to the

[1]https://github.com/sanajavaid01/codevid-datasets


stackoverflow
Crawling stackoverflow for programming related questions
Programming expert to filter relevant questions
Computer and Programming related Questions
Programming instructional questions
YouTube
Querying YouTube for relevant Videos
Annotation Process:Expert to Formulate the Question and Mark the Answer Timestamps in the Video
CodeVidQA Dataset
HEIGHT WIDTH
IN 6 MINUTES!
FETCH
fetch()
PAGINATION
IN 8 MINUTES!
DIGITAL CLOCK
03:07:44 PM
ASYNC / AWAIT
.then()
await
WEATHER APP
Miami
90°F

Questions
Why is processing a sorted array faster than processing an unsorted array?
How do I undo the most recent local commits in Git?
How do I delete a Git branch locally and remotely?
What is the difference between 'git pull' and 'git fetch'?
What does the "yield" keyword do in Python?
How can I remove a specific item from an array in JavaScript?

Questions	Category
Why is processing a sorted array faster than processing an unsorted array?	Non-Programming
How do I undo the most recent local commits in Git?	Programming Instructional
How do I delete a Git branch locally and remotely?	Programming Instructional
What is the difference between 'git pull' and 'git fetch'?	Programming Instructional
What does the "yield" keyword do in Python?	Programming Instructional
How can I remove a specific item from an array in JavaScript?	Programming Instructional

Video ID	Programming Question	Answer Start Timestamp	Answer End Timestamp
afdfsfwe44d	How to implement queues in laravel?	00:08	00:38
3fgbvfgf55t	What is purpose of Fragment in reactjs?	04:32	05:31
Fgbttvh56j6	How to implement migration in laravel?	01:05	02:15
...	...	...	...
G6frtdu7kj	How to implement seeds in laravel?	01:01	01:31

Figure 1: Schematic Overview of the Methodology for CodeVidQA Dataset

model development.

Step 1: Crawling Stack Overflow for programming questions.

Step 2: Filtering programming instructional and non-programming videos.

Step 3: Querying YouTube for relevant videos.

Step 4: Annotation Process: The expert formulates the question and marks the answer timestamps in the video.

Figure 1 shows the steps of workflow for creation of CodeVidQA Dataset.

3.1.1 Extracting Stack overflow questions

Dataset generation begins with the collection of programming-related questions from community question-answering (CQA) platforms such as Stack Overflow. As one of the largest CQA sites, Stack Overflow comprises millions of questions posted by developers.

We collected about 2,500 high-voted questions in different programming languages such as HTML, JavaScript, and Python using the Stack Overflow v2.3 API[2] without limiting by tags.Figure 2 shows the tag cloud of the tags against questions extracted from tack Overflow API. Most frequently occurring tags are JavaScript and Python which are shown in larger font depicting the large number of questions belong a specific language. These extracted questions act as topics for searching instructional videos.

[2]https://api.stackexchange.com/2.3/questions

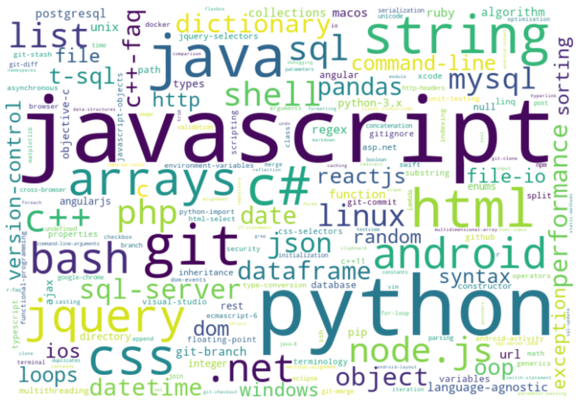


Figure 2: Tag Cloud of frequently used tags against Stack Overflow questions

3.1.2 Identification of relevant programming-related questions

High-voted questions were collected from Stack Overflow using the API and then identified as either programming instructional or non-programming instructional. Questions like “Daylight saving time and time zone best practices” or “What is the difference between POST and PUT in HTTP?” are considered as non-instructional because they can be demonstrated through theory without any implementation. The task of categorization of the questions was done manually by the programming experts and they divided them into 1,946 ‘programming instructional’ and 553 ‘non-programming instructional’. Figure 3 illustrates the distribution, questions with tags JavaScript and Python accounting for 16.9% and 14.3%, and the questions with tags Java and Git making up a part of 11% each.

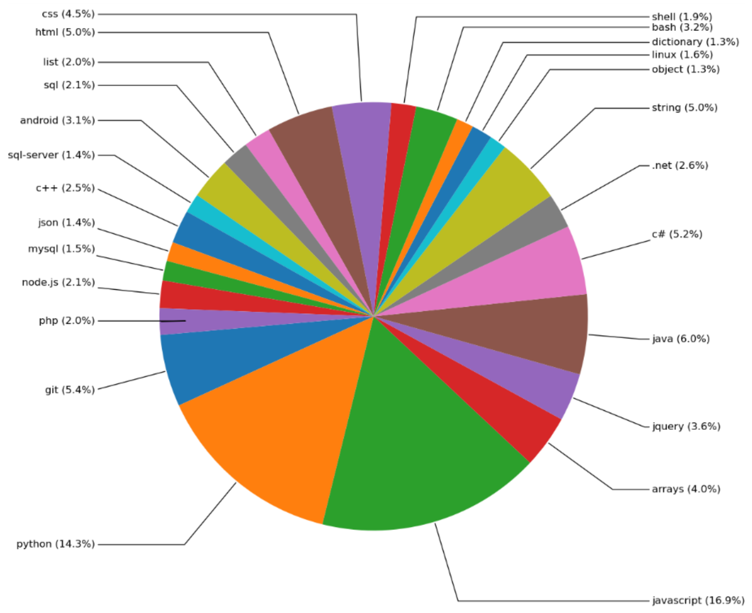


Figure 3: Distribution of the instructional programming questions category collected from Stack overflow

3.1.3 Searching YouTube Videos Relevant to Each Programming Instructional Question

The next important process in dataset construction is the process of collecting corresponding YouTube videos associated with a particular Stack Overflow question or topic. The following criteria were used for selecting videos:

i. There have to be at least a thousand subscribers of a channel from which the video will be selected.

ii. Videos should be in English since the majority of tutorials are in English (80%) then German (4%) and Spanish (2%) (Kadriu et al., 2020).

iii. Subtitles should be on and are in the English language.

iv. If a video does not contain a verbal explanation it is considered non-instructional.

v. The Integrated Development Environment(IDE) has to be used for the implementation should be observable.

For each question or topic, a YouTube search option is performed with the most relevant high-quality video. Selection is made on CONTENT only, PRESENTER's way of teaching and ANSWERS provided are all from the first page results.

3.1.4 Expert Annotation for Programming Instructional Videos

Programming experts need to identify videos as either "programming instructional" or "programming non-instructional" based on a YouTube search. This important step is necessary because:

- Retrieve videos from the search may contain theoretical communication rather than programming against a specific programming query.

- It is essential to ensure the reliability of programming videos as instructional.

To identify a programming video as instructional from the pool of YouTube videos, the following criteria should be met:

i) Programming instructional videos should demonstrate the implementation of a specific problem that is queried using YouTube search, yielding decent results for the problem.

ii) Programmers should clearly define each step of the implementation using an Integrated Development Environment (IDE) and provide reasoning for each step.

Programming instructional videos can be of any level of expertise. Most programmers begin the video with an introduction and end with concluding notes. Skip these sections while annotating videos. Only the segment of the video where the programmer provides proper steps for implementation considers the answer.

3.1.5 Formulating Instructional Questions and Visual Answers from Videos

After defining what programming instructional videos are, the following steps were followed to generate programming-related questions and mark their answers. A programming expert marked the starting and ending timestamps of the answer and formulated the question for annotated timestamps along with the identification of the programming language.Of course, the majority of questions were about implementation, But basic notes or explanations also apply to the videos, for instance, "How do I align an element to the center in the horizontal direction?"

For the majority of the cases, some questions were either different or were of the same timestamps whereby some of the tutorials provide around 15 questions. This approach serves to make questions diverse to accommodate how developers approach a problem only to find the same resulting answers or solutions.

3.1.6 Creating dataset having a question and respective answers

After formulating questions against timestamps, the programming expert has to add the video Id, question, starting timestamp, and ending timestamp along with the programming language of the video in an excel sheet. This resulted in the creation of 2,104 pairs of question and answers timestamps. The programming language feature will help the video question-answering system to search videos against a specific language if the user mentions that language in its query.

3.2 CodeVidCL Creation

A programming-based video (QA) system that aids visual answers to programming-related questions must be able to distinguish between "programming" and "non-programming" videos. For building these types of systems that can perform effectively on datasets CodeVidCL needs to be created that can train the system to differentiate between programming and non-programming videos.

In the first step, a video classifier will be trained that can be utilized to get a high-confidence video category. In the second step, programming experts validated the programming and non-programming video categories predicted by the classifier and then sampled those videos for the CodeVidCL dataset.

The following steps should be performed to achieve the proposed solution for the creation of CodeVidCL. Figure 4 shows the steps of workflow follow for the creation of CodeVidCL Dataset.

Step 1: Collecting programming and non-programming videos.

Step 2: Extracting subtitles of videos.

Step 3: Build an ensemble classifier based on Big Bird transformers.

Step 4: Get the high confidence of the video category.

Step 5: Validate predicted video categories and add them to the dataset.

3.2.1 Collecting programming and non-programming videos

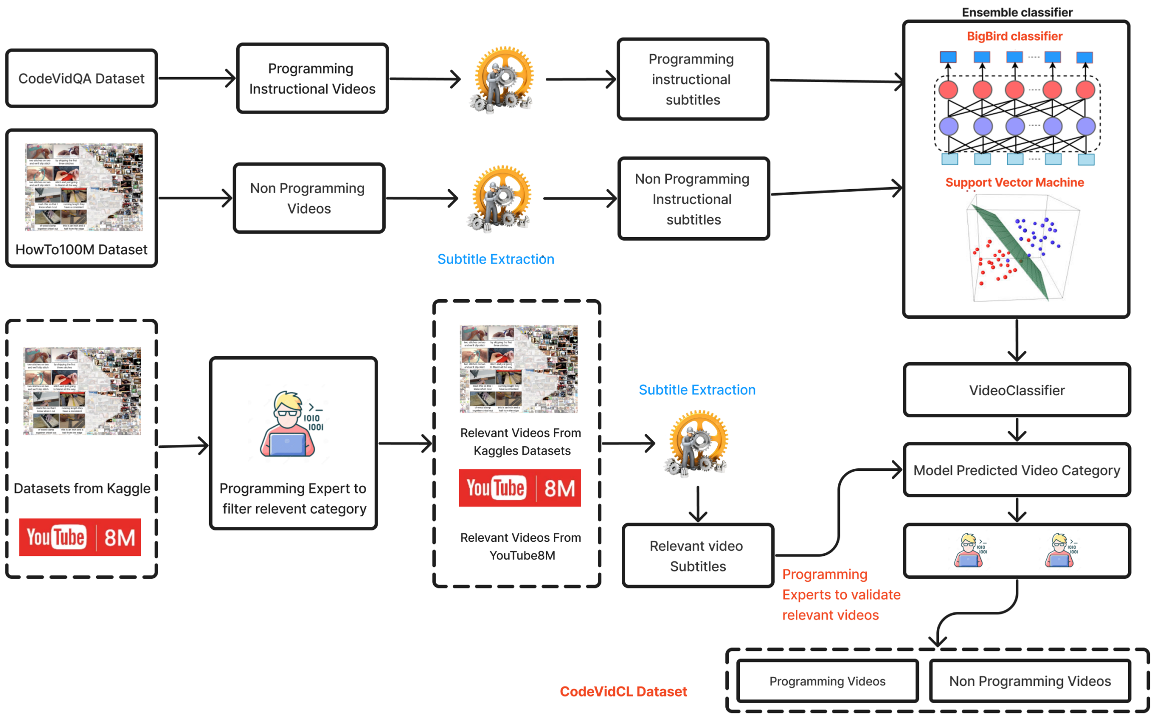
To train the classifier, we need to collect programming and non-programming videos for the training dataset that can be input for fine-tuning in the classifier. We utilized 2,104 human-annotated programming videos from the CodeVidQA dataset and for non-programming videos, we sampled 3,795 videos from HowTo100M (Jang et al., 2017), which is a large-scale YouTube dataset based on instructional videos from various categories like food, art, craft, sports, cars and vehicles etc. There are total of 143 categories in HowTo100M data having 12,38,912 entries. Figure 5 shows number of entries against the top 20 categories. To remove the imbalance between programming and non-programming videos, the HowTo100M dataset was reduced to get the equally distributed entries against each category that resulted in 27 entries for each category except Diwali(26 entries), School Stuff(22 entries), Social Activism(15 entries)and National Days(6 entries) categories.

3.2.2 Extracting subtitles for all video

A total of 6,104 collected videos in previous are taken and pass to YouTube API to get subtitles. We use the YouTube-transcript-API[3] module of Python to get subtitles of video along with many words.

Most of the videos have to disable transcripts or use a language other than English for transcribing. This type of video is eliminated from the dataset based on the number of words. After doing all the data cleaning, add one more column in the Python dataframe with name category (programming, non-programming) and class (1, 0).Using a stratified

[3]https://pypi.org/project/youtube-transcript-api/



CodeVidQA Dataset
Programming Instructional Videos
Programming instructional subtitles
HowTo100M Dataset
Non Programming Videos
Non Programming Instructional subtitles
Subtitle Extraction
Ensemble classifier
BigBird classifier
Support Vector Machine
Datasets from Kaggle
Programming Expert to filter relevent category
Relevant Videos From Kaggles Datasets
Relevant Videos From YouTube8M
Subtitle Extraction
Relevant video Subtitles
Programming Experts to validate relevant videos
VideoClassifier
Model Predicted Video Category
CodeVidCL Dataset
Programming Videos
Non Programming Videos

Figure 4: Diagrammatic Representation of the Methodology for CodeVidCL Dataset

Table 1: Performance metrics for different models

Models	Accuracy	Precision	Recall	F1 Score
BigBird (bigbird-roberta-large)	99.61%	99.05	99.01	99.50
SVM	99.80%	99.81	99.81	99.81

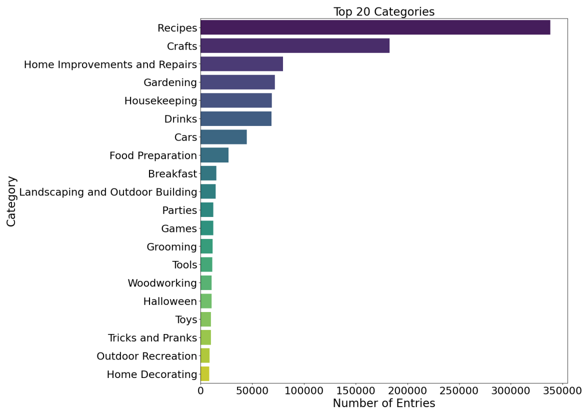


Figure 5: Number of Entries against top 20 Categories of HowTo100M

splitting, we utilized 20 % for testing, and 80 % of the videos in the collection for training purposes.

3.2.3 Building Video Classifier

In the next step, a classifier is trained on training dataset videos that are created in the previous step. The subtitles are first extracted here because they can be longer than 2000-3000 words. A classifier based on a transformer, the BigBird(Zaheer et al., 2020) model by Hugging Face [4] and SVM are trained. SVM was used as a statistical classifier that is effective for categorization tasks, especially when the data is structured and separable into distinct categories, which fits well for coarse-grained categorization.BigBird was chosen as a deep learning classifier due to its capability to handle long-range dependencies and accommodate long sequences like subtitles from videos. BigBird is suitable for capturing context from the sequential nature of subtitles, which is common in videos. The models above were combined in the ensemble classifier, and majority voting is apply in making the final forecasts. The subtitles were extremely lengthy and hence, it was crucial to work with large-grained categories such as programming and non-programming.

BigBird is designed for sequence processing and training on PyTorch. To combat class imbalance, class weights were modified, early stopping was

[4]https://huggingface.co/google/bigbird-roberta-large

used and a variable learning rate was implemented. Training settings were warm-up scheduler with 5 steps, weight decay of 0.0001, gradient accumulation with 4 steps and mixed precision activated by fp16. This model was trained for one epoch with a batch size of four and was evaluated every 20 steps.

Labels in the SVM were represented numerically, and the TF-IDF was used as features and balanced class to address the issue of imbalance. Predictive probabilities of the class distribution were computed and the final Classifier uses votes between BigBird and SVM predictions for accurate classification. This approach enhances the performance of these models making them a hybrid. Table 1, shows the accuracies of fine-tuned BigBird and SVM.

3.2.4 Identification of relevant video

We selected a subset of YouTube8M's(Abu-El-Haija et al., 2016) computer-related, non-programming videos as well as other Kaggle datasets that have been expertly categorized by programmers.

We built a subset from YouTube8M and different Kaggle datasets, such as YouTube data science[5] and YouTube programming videos from free code camp[6], TED-ED[7], and caption datasets[8], after training our ensemble model. 5,722 videos were produced overall from this procedure, of which 3,120 were classified as programming and 2,602 as non-programming.

3.2.5 Predicting Relevant Video Category Using Video Classifier

In this step, we utilized an ensemble setting for predicting videos with a high confidence vote on the category for the dataset gathered in the previous step. The ensemble classifier predicted 1,751 programming videos and 2,540 non-programming videos. To create a high-quality dataset, programming experts only chose those videos to which the classifier gives high confidence in the category.

3.2.6 Sampling High-Quality Video

In the last step, videos with a predicted category and high confidence are validated and chosen by programming experts to create a high-quality CodeVidCL dataset. This dataset contains the video title, video Id, video category, subtitle, and number of words in subtitles for each video.

4 Results & Analysis

4.1 CodeVidQA Analysis and Validation

When building CodeVidQA, we aim to compile reliable programming courses from YouTube only. This way we can say that a video is reliable if it has more than a thousand subscribers, from a reliable programming institute, from a famous programmer or famous programming platforms like W3Schools, Treehouse, etc. We collected 2,500 questions about programming from Stack Overflow, selecting only those that contain the instructions that can be illustrated in an IDE. Such theoretical questions as those which do not require an answer in instructional situations were not included. The second phase only included the instructional questions, therefore, we have 2,104 paired questions and visual answers obtained from 1, 363 instructional videos, making up 132 hours of video. Figure 7 and Figure 8 show the answer duration of videos in seconds and the distribution of question length in CodeVidQA dataset respectively. For Python alone, more than 400 pairs were produced and this was trailed by both JavaScript and Flutter. Figure 6 shows the number of question answers key pairs against programming language.

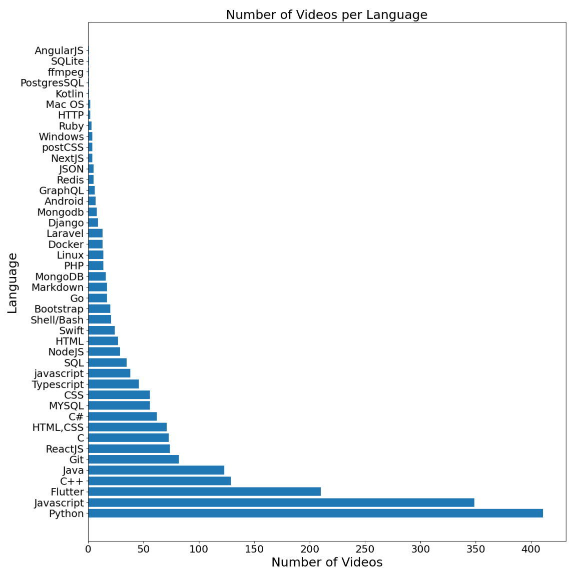


Figure 6: Number of videos against each programming language in CodeVidQA

For evaluation,100 questions are sampled that

[5]https://www.kaggle.com/datasets/sandhyakrishnan02/youtube-datascience-video-views

[6]https://www.kaggle.com/datasets/nuhmanpk/all-programming-tutorial-from-free-code-camp

[7]https://www.kaggle.com/datasets/hadilhagar/ted-ed-dataset-acquired-via-youtube-api

[8]https://www.kaggle.com/datasets/shivendrra/youtube-caption-dataset-for-finetuning-or-training

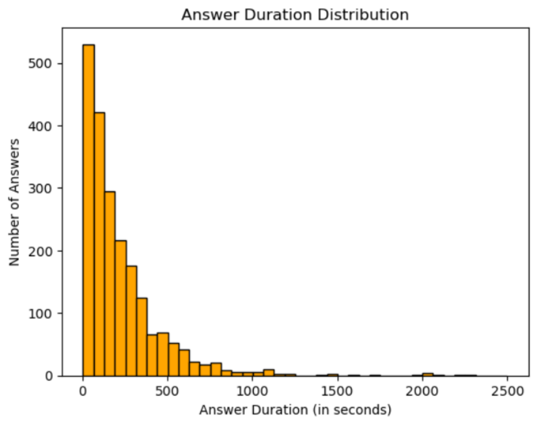


Figure 7: Answer Duration Distribution of CodeVidQA

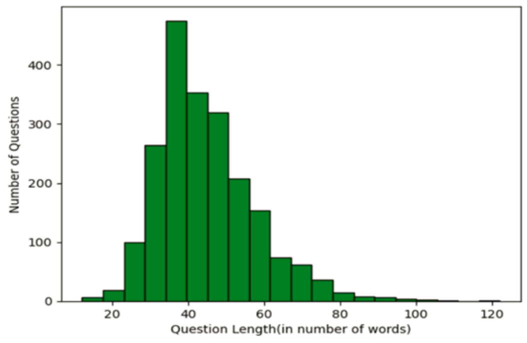


Figure 8: Question Length Distribution of CodeVidQA

are extracted from Stackoverflow and two programming experts categorize them as instructional or non-instructional. After categorizing those both programmers are in same agreement except for a few questions upon which theoretical and practical demonstration can be given. To validate the dataset, 50 videos are sampled and ask both programming experts to annotate answers along with the formulation of questions. Semantic similarity of formulated questions were assessed and the absolute differences between answers timestamped were calculated. The second assessment validates their agreement on proving the precise and valid answer timestamps from the videos. We found that both the annotators formulated 80 and 67 questions, and 54 out of them were semantically similar. The CodeVidQA dataset's quality is confirmed by these evaluations.

4.2 CodeVidCL Analysis and Validation

To construct the CodeVidCL dataset, we took a selection of human-annotated programming instructional videos from the CodeVidQA dataset and non-programming videos from HowTo100M dataset that is used as the training set for the CodeVidCL dataset. To generate a validation and test set, we sampled high-confidence videos that a video classifier had predicted. To further evaluate CodeVidCL dataset, we asked both the programming experts to look over the video category that the model had predicted on a dataset created from YouTube8M and different Kaggle datasets .

The experts asked to update the category and label videos as non-relevant if there is insufficient data to categorize them into any of the categories in case the video classifier incorrectly classifies them. The final CodeVidCL collection contains 4291 videos, 2540 of which are non -programming videos and the remaining ones are programming.

5 Conclusion

One of the most famous and actively developing methods of obtaining knowledge is the use of online instructional videos, especially in programming. Video Question Answering (VideoQA) is an essential research domain, that focuses on equipping AI with the ability to interpret and engage with visual information using natural language. However, VideoQA is significantly less investigated than Image-QA, which creates difficulties for models to understand the content of videos and to answer the queries. To fill these gaps, this research proposes two datasets, CodeVidQA and CodeVidCL for programming instructional video question answering and classification.

CodeVidQA contains 2,104 entries where each entry represents an expert-curated programming question-answers pair along with a video Id. Moreover, up to 99.61% accuracy has been achieved after the training of the BigBird model and SVM for the creation of the CodeVidCL dataset. These results not only prove the applicability of the proposed datasets but also open the research avenue for the enhancements of future VideoQA systems regarding the preciseness of response and richness of the learning experience in programming through videos.

6 Limitations

The existing dataset covers only a limited set of programming languages for video classification and question answering. A larger and more diverse dataset, such as CodeVidQA and CodeVidCL, which spans a broad range of programming topics, is needed. Additionally, result explainability is a significant factor, as current models do not identify the features that contribute to their predictions.

References

Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. 2016. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*.

Joel Brandt, Philip J Guo, Joel Lewenstein, Mira Dontcheva, and Scott R Klemmer. 2009. Two studies of opportunistic programming: interleaving web foraging, learning, and writing code. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1589–1598.

Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, et al. 2020. Proceedings of the twelfth language resources and evaluation conference. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*.

Seongho Choi, Kyoung-Woon On, Yu-Jung Heo, Ahjeong Seo, Youwon Jang, Minsu Lee, and Byoung-Tak Zhang. 2021. Dramaqa: Character-centered video story understanding with hierarchical qa. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 1166–1174.

Sergio Guadarrama, Niveda Krishnamoorthy, Girish Malkarnenkar, Subhashini Venugopalan, Raymond Mooney, Trevor Darrell, and Kate Saenko. 2013. Youtube2text: Recognizing and describing arbitrary activities using semantic hierarchies and zero-shot recognition. In *Proceedings of the 2013 IEEE International Conference on Computer Vision*, ICCV '13, page 2712–2719, USA. IEEE Computer Society.

Deepak Gupta, Kush Attal, and Dina Demner-Fushman. 2023. A dataset for medical instructional video classification and question answering. *Scientific Data*, 10(1):158.

Thierry Hamon, Natalia Grabar, and Fleur Mougin. 2017. Querying biomedical linked data with natural language questions. *Semantic Web*, 8(4):581–599.

Phil Hill. 2014. Online educational delivery models: A descriptive view.

Yunseok Jang, Yale Song, Youngjae Yu, Youngjin Kim, and Gunhee Kim. 2017. Tgif-qa: Toward spatio-temporal reasoning in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2758–2766.

Arbana Kadriu, Lejla Abazi-Bexheti, Hyrije Abazi-Alili, and Veland Ramadani. 2020. Investigating trends in learning programming using youtube tutorials. *International Journal of Learning and Change*, 12(2):190–208.

Khushboo Khurana and Umesh Deshpande. 2021. Video question-answering techniques, benchmark datasets and evaluation metrics leveraging video captioning: a comprehensive survey. *IEEE Access*, 9:43799–43823.

Kyung-Min Kim, Min-Oh Heo, Seong-Ho Choi, and Byoung-Tak Zhang. 2017. Deepstory: Video story qa by deep embedded memory networks. *arXiv preprint arXiv:1707.00836*.

Jie Lei, Licheng Yu, Mohit Bansal, and Tamara L Berg. 2018. Tvqa: Localized, compositional video question answering. *arXiv preprint arXiv:1809.01696*.

Jie Lei, Licheng Yu, Tamara L Berg, and Mohit Bansal. 2019. Tvqa+: Spatio-temporal grounding for video question answering. *arXiv preprint arXiv:1904.11574*.

Vanessa Lopez, Victoria Uren, Enrico Motta, and Michele Pasin. 2007. Aqualog: An ontology-driven question answering system for organizational semantic intranets. *Journal of Web Semantics*, 5(2):72–105.

Jonghwan Mun, Paul Hongsuck Seo, Ilchae Jung, and Bohyung Han. 2017. Marioqa: Answering questions by watching gameplay videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2867–2875.

Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2016. Movieqa: Understanding stories in movies through question-answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4631–4640.

Dejing Xu, Zhou Zhao, Jun Xiao, Fei Wu, Hanwang Zhang, Xiangnan He, and Yueting Zhuang. 2017. Video question answering via gradually refined attention over appearance and motion. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1645–1653.

Weijiang Yu, Haoteng Zheng, Mengfei Li, Lei Ji, Lijun Wu, Nong Xiao, and Nan Duan. 2024. Learning from inside: self-driven siamese sampling and reasoning for video question answering. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA. Curran Associates Inc.

Zhou Yu, Dejing Xu, Jun Yu, Ting Yu, Zhou Zhao, Yueting Zhuang, and Dacheng Tao. 2019. Activitynet-qa: A dataset for understanding complex web videos via question answering. *Preprint*, arXiv:1906.02467.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297.