Visual Question Answering (VQA) Simpsons Challenge

Brandon Smith
s222139956@deakin.edu.au
Deakin University
Melbourne, Australia

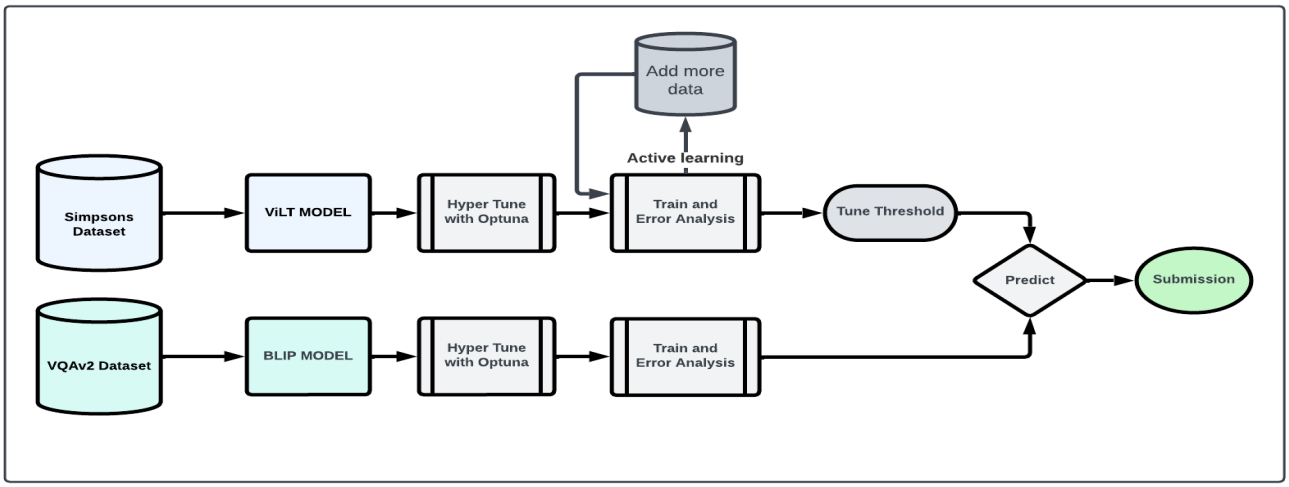

Figure 1. High-level overview of training implementation.

Abstract

This paper provides a clear guide for newcomers to Machine Learning and Visual Question Answering (VQA), focusing on the Deakin Simpsons Challenge. We detail our approach, highlighting successful techniques, reflecting on less effective methods, and providing key insights for future competitors. Techniques discussed include active learning for dataset creation, hyperparameter optimisation using Optuna, the use of ensemble learning with pre-trained models, and fine-tuning of the prediction threshold for model optimisation. This paper aims to explain the process, offering practical insights for future competition participants.

Keywords: Transfer learning, computer vision, visual question answering, hypertuning, active learning, ensemble

Deakin Simpsons Challenge, Deakin University, 2023

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/XXXXXXX.XXXXXXX

ACM Reference Format:
Brandon Smith. 2023. Visual Question Answering (VQA) Simpsons Challenge. In *Proceedings of Deakin Simpsons Challenge.* ACM, Melbourne, Vic, Australia, 5 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

The Deakin Simpsons Challenge presents a fun challenge: given an image of characters from The Simpsons and a question about the image, how can we generate an accurate response using machine learning and deep learning techniques? This challenge is addressed through the development of a Visual Question Answering (VQA) model [5].

VQA is a fusion of visual understanding and language processing, requiring the computer to assimilate both text features (derived from the question) and visual features (derived from the image) to formulate an answer. In other words, VQA is the process of teaching a computer to comprehend an image and a related question, and to provide an answer in natural language [2].

From here, we will dive into the methodology that achieved the highest accuracy for the challenge. The discussion starts with the training datasets used, the implementation of the pre-trained models, the optimisation of hyperparameters using the Tree-structured Parzen Estimator (TPE) algorithm, the application of active learning, fine-tuning the prediction threshold and the process of ensembling models.

While not introducing novel concepts or tools, this paper provides a comprehensive case study of effective strategy development in tackling a specific problem using existing tools and resources. The objective is to demonstrate how these methods can be leveraged in similar scenarios, thereby aiding future endeavors in the world of Machine Learning competitions. The implementation is available at https://github.com/brandonsmith301/deakin-ai-challenge.

2 Previous Winner

Looking into the successful strategies of the 2022 Deakin Simpsons Challenge winner [9] offered valuable insights for the challenge, showcasing effective machine learning methods like transfer learning. Acknowledging past attempts in the same problem domain is not a matter of imitation, but a means of learning from prior experiences.

Each foray into the field of Machine Learning serves as an informative stepping stone, shaping the trajectory of future attempts [13]. This is the essence of iterative progression in the field, enabling us to stand on the shoulders of those who have gone before, saving time and refining our efforts.

3 Training Data

The fine-tuning process depend on the utilisation of two distinctive datasets, corresponding to each model, as depicted in Figure 1. For the BLIP model, the Binary VQAv2 dataset [15] was used, a well-recognised and widely used resource in this field containing 20,629 images, 22,055 questions and 220,550 answers. The ViLT model was fine-tuned using a newly compiled Simpsons dataset, a collection featuring over 7000 unique images and 10,000 binary-answer questions. These Simpson images were sourced from Frinkiac [10].

4 Pre-trained Models

In pursuit of the "most" optimal pre-trained ML model, we explored a few other options, acknowledging the absence of a single "best" model as per the No Free Lunch theorem [8]. The testing revealed that among the various models considered, two emerged as top performers.

- BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation [12].
- ViLT: Vision-and-Language Transformer Without Convolution or Region Supervision [11].

The pre-trained models were utilised through Hugging Face Transformers, which offers convenient APIs and tools for downloading and training state-of-the-art pre-trained models [1]. This transfer learning approach inspired by the work of the previous winner proved invaluable in the work for the Deakin Simpsons Challenge, as it saved us a substantial amount of time that would have been otherwise spent on designing and extensively training a model from scratch.

4.1 BLIP Model

The BLIP model as explained in the paper [12] key components are:

- **Visual Transformer:** Divides an image into patches and converts these patches into a sequence of numbers.
- **Unimodal Encoder:** Independently encodes image and text, with the text encoder similar to the BERT model.
- **Image-grounded Text Encoder:** Encodes both text and visual information, connecting them using a cross-attention layer.
- **Image-grounded Text Decoder:** Generates text based on an image using causal self-attention layers.

4.2 ViLT Model

The ViLT model discussed in the paper [11]. This is the same model used by the winner for the Deakin Simpsons Challenge in 2022 [9]. Key aspects of the ViLT model are:

- **Initialisation:** Initialises transformer weights from a pre-trained Visual Transformer (ViT).
- **Text Embedding:** Embeds input text using a word embedding matrix and a position embedding matrix.
- **Image Embedding:** Divides input image into patches, each converted into interpretable form and embedded with position information.
- **Combining Text and Image Embeddings:** Sums text and image embeddings with their corresponding modal-type embedding vectors, then concatenates them into one sequence.
- **Contextualising the Vector:** Contextualises the data by updating the combined sequence through multiple layers of the transformer model.
- **Pooled Representation:** Creates a pooled representation of the entire multimodal input by applying a linear projection and a hyperbolic tangent function on the first index of the final sequence.

4.3 Key Differences

ViLT is faster and easier to train because it simplifies the process of integrating textual and visual information by simply summing and concatenating the embeddings, therefore reducing computational complexity [11].

On the other hand, the BLIP model, although it may be more computationally intensive and time-consuming to train, achieves higher accuracy, compared in Table 2 and 3. Instead of simply summing and concatenating the embeddings, BLIP uses a dedicated Image-grounded Text Encoder to connect visual and textual information using a cross-attention layer. [12].

Table 1. Fine-tuning details for the BLIP and ViLT models.

Model	Epochs	Dataset	Image Size	Batch Size	Learning Rate	Weight Decay
BLIP	0.5	Binary VQAv2	256×256	2	2×10^{-5}	0.05
ViLT	4	Simpsons Dataset	256×256	64	4.81×10^{-5}	0.071

5 Fine-tuning Details

As shown in Table 1, the two different models were fine-tuned on two different datasets. Initially, the training approach of freezing the weights and only unfreezing the last layer of each of the models was tested. However, this approach did not yield successful results. Consequently, all the weights were fine-tuned.

Both models utilised an AdamW optimiser. However, the BLIP model differed in its use of a learning rate scheduler, employing a cosine schedule with hard restarts and warmup.

6 Active Learning

Mid-way through the competition, an active learning-inspired approach was employed, although it differed slightly from traditional active learning methodologies . Rather than querying for labels on uncertain instances [6], this approach involved conducting an error analysis post-training.

Specifically, after training the ViLT model, an error analysis was performed to identify the areas where the model struggled the most. Once these areas were identified, new data was created to specifically address these challenging aspects.

This approach diverges from classical boosting in an important way: instead of creating an ensemble of models to improve on the errors of the previous ones [7], the strategy here was to iteratively improve a single model (in this case, ViLT) by effectively targeting its areas of weakness with specialised training data.

7 Hyperparameter Tuning

We employed Optuna, an automatic hyperparameter optimisation software framework specifically designed for machine learning [3]. The Optuna framework facilitates the efficient optimisation of hyperparameters by offering multiple tuning strategies. For our models, we opted for the Tree-Structured Parzen Estimator (TPE), which uses a Bayesian optimisation method for hyperparameter tuning. The hyperparameters tuned were the learning rate and weight decay.

We configured the TPE sampler with three startup trials, which refers to the initial number of random explorations of the hyperparameter space. This approach is beneficial as it allows the algorithm to balance exploration and exploitation from the very start [4]. Algorithm 1 presents a very basic overview of how TPE sampler works, however, for a more detailed explanation and understanding, refer to the work cited in [14].

Algorithm 1: Tree-Structured Parzen Estimator (TPE) - High-level Overview

Result: Hyperparameters in T that resulted in the minimum loss in L
Phase 1: Initialisation;
Number of iterations, N;
Number of startup trials, S;
Initialise an empty list of trials, T;
Initialise a list of losses, L;
Phase 2: Iterative Sampling;
for *iteration* i *from* 1 *to* N **do**
  if $i < S$ **then**
    Randomly sample hyperparameters, h;
  else
    Build a probability model, P, based on past trials and their losses;
    Sample hyperparameters, h, that are expected to minimise loss according to P;
  end
  Run a trial with the selected hyperparameters, h, and record the loss, l;
  Add h to the trial list, T, and l to the loss list, L;
end
Phase 3: Result Compilation;
Return the set of hyperparameters that resulted in the lowest loss;

8 Threshold Tuning

In response to observed correlations between precision and accuracy, the ViLT model was specifically fine-tuned using ROC AUC threshold tuning. This was motivated by the model's sensitivity to correct positive instance classification, significantly impacting precision and thus accuracy.

Using the ROC AUC, an optimal threshold was identified to balance sensitivity and specificity, aiming for maximal precision. This process facilitated the ViLT model to classify positive cases optimally, enhancing both precision and accuracy effectively

9 Ensembling

An ensemble method was used to combine the predictions of the BLIP and ViLT model. Specifically, a logical disjunction "OR" was used.

The final prediction was derived from the outputs of both models. If P_1 and P_2 are the predictions of Model 1 and Model 2 respectively, the final ensemble prediction P_e is computed as follows:

$$P_e = P_1 \vee P_2$$

Various ensemble methods were tested, but the logical OR operation yielded the highest accuracy. An alternative approach that showed promise involved averaging the logits of multiple ViLT models before executing the logical OR operation. However, this method did not surpass the performance of the straightforward logical OR operation between the predictions from the BLIP and ViLT models.

10 Results

Table 2. ViLT Test Accuracy

Steps	ViLT Accuracy (%)
Base	53.6
9000	52.4
10000	56.0
15000	56.4
20000	55.4

Table 3. BLIP Test Accuracy

Steps	BLIP Accuracy (%)
Base	55.2
1000	54.8
1500	57.2
2000	56.4
2500	56.4

The results presented above represent the performance of both ViLT and BLIP models after extensive hyperparameter tuning. In terms of base accuracy (i.e., without any fine-tuning), the BLIP model surpasses ViLT, clocking in at 55.2 compared to ViLT's 53.6. This suggests that BLIP offers a more accurate base model.

The varying step counts between the two models are due to the training efficiency of each model. The ViLT model exhibits impressive training efficiency, allowing for higher step counts (up to 20,000 steps), while maintaining accuracy. Although the BLIP model's training process is slower, this has not hindered its performance. In fact, it has demonstrated higher accuracy with fewer steps, reaching an accuracy of 57.2 at 1500 steps.

Table 4. Ensemble Test Accuracy

Method	Ensemble Accuracy (%)
10000 ViLT + 1500 BLIP	57.2
15000 ViLT + 1500 BLIP + AL	58.8
25000 ViLT + 1500 BLIP + AL + TT	59.6

The Ensemble model combines the ViLT and BLIP models for enhanced performance. The configuration where ViLT is trained to 10,000 steps and BLIP to 1,500 steps results in an accuracy of 57.2. By further training ViLT to 15,000 steps and implementing Active Learning (AL) with the BLIP model at 1,500 steps, the accuracy increases to 58.8.

The highest accuracy of 59.6 is achieved when ViLT is trained to 25,000 steps, and both Active Learning (AL) and Threshold Tuning (TT) are applied to the 1,500 step BLIP model.

11 Recommendation

To wrap up this paper, we think it's fitting to share some lessons learned from our experience with the Deakin Simpsons Challenge 2023. Think of them not as a surefire recipe for winning next year's competition, but as practical advice that could elevate your game and make you a monumental participant.

Firstly, start by creating a high-quality training dataset that mirrors the problem you're tackling. A model is only as good as the data it learns from, and having data that reflects your specific challenge can significantly boost your model's accuracy. We realised the hard way that we should have spent less time trying to perfect a model based on the given template and more time creating a relevant, quality dataset. This task can be more efficiently and effectively accomplished in a team.

Secondly, don't reinvent the wheel; use transfer learning. Building a model from scratch not only consumes a lot of time but also may not perform as well as the existing state-of-the-art models. Utilise what's already proven successful and build upon it.

Thirdly, hyperparameter tuning is not to be neglected. Consider adjusting parameters such as batch size, learning rate, and weight decay. In our case, employing stratified k-fold during training to combat data imbalance didn't yield significant improvements, but it may be a worthwhile strategy in different circumstances.

Lastly, consider fine-tuning the threshold of your model and ensembling your models for the final prediction. Although a quality dataset will likely be the primary driver of success, these strategies can add an extra boost to your model's performance.

While our recommendations aren't a guaranteed blueprint for winning, they are practical tips forged from our experience with the competition. For a more in-depth guide, we do recommend "How to avoid machine learning pitfalls: a guide for academic researchers" by Michael A. Lones [13]. The cited paper, provides a deeper discussion for each recommendation.

References

[1] [n. d.]. Hugging Face. https://huggingface.co/.

[2] Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. Lawrence Zitnick, Dhruv Batra, and Devi Parikh. 2015. VQA: Visual Question Answering. *arXiv preprint arXiv:1505.00468* (2015). https://doi.org/10.48550/arXiv.1505.00468

[3] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A Next-generation Hyperparameter Optimization Framework. (2019). https://doi.org/10.48550/arXiv.1907.10902 arXiv:1907.10902 [cs.LG] 10 pages, Accepted at KDD 2019 Applied Data Science track.

[4] J. Bergstra, D. Yamins, and D. D. Cox. 2013. Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures. *Rowland Institute at Harvard* (2013). Emails: bergstra@rowland.harvard.edu, yamins@mit.edu, davidcox@fas.harvard.edu.

[5] Mohamed Reda Bouadjenek. 2023. *Deakin Simpsons Challenge 2023*. Retrieved July 22, 2023 from https://rbouadjenek.github.io/deakin-ai-challenge2023/

[6] Wikipedia contributors. 2023. Active learning (machine learning). https://en.wikipedia.org/wiki/Active_learning_(machine_learning). Accessed on July 13, 2023.

[7] Wikipedia contributors. 2023. Boosting (machine learning). https://en.wikipedia.org/wiki/Boosting_(machine_learning). Accessed on July 13, 2023.

[8] Wikipedia contributors. 2023. No free lunch theorem. https://en.wikipedia.org/wiki/No_free_lunch_theorem. Accessed on July 13, 2023.

[9] Jessie Xiaojuan He. [n. d.]. VQA on Simpsons scenes: Transfer learning from pre-trained Vision-and-Language Transformer. ([n. d.]).

[10] Paul Kehrer, Sean Schulte, and Allie Young. 2016. Frinkiac. https://frinkiac.com/.

[11] Wonjae Kim, Bokyung Son, and Ildoo Kim. 2021. ViLT: Vision-and-Language Transformer Without Convolution or Region Supervision. *arXiv preprint arXiv:2102.03334* (2021). https://doi.org/10.48550/arXiv.2102.03334 ICML 2021 Long Presentation, Submitted on 5 Feb 2021 (v1), last revised 10 Jun 2021 (this version, v2).

[12] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. 2022. BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation. *arXiv preprint arXiv:2201.12086* (2022). https://doi.org/10.48550/arXiv.2201.12086 Submitted on 28 Jan 2022 (v1), last revised 15 Feb 2022 (this version, v2).

[13] Michael A. Lones. 2023. How to avoid machine learning pitfalls: a guide for academic researchers. (2023). https://doi.org/10.48550/arXiv.2108.02497 arXiv:2108.02497 [cs.LG] Comments: 25 pages.

[14] Shuhei Watanabe. 2023. Tree-Structured Parzen Estimator: Understanding Its Algorithm Components and Their Roles for Better Empirical Performance. (2023). https://doi.org/10.48550/arXiv.2304.11127 arXiv:2304.11127 [cs.LG]

[15] Peng Zhang, Yash Goyal, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2016. Yin and Yang: Balancing and Answering Binary Visual Questions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.