Engineering and Applied Science Programs for Professionals
Whiting School of Engineering
Johns Hopkins University
685.701 Data Science: Modeling and Analytics Programming Assignment 2 v2
Building Custom Models with Hugging Face for the Flickr30k Dataset
Assigned with Module 8
Due at the end of Module 14

Total Points 100/100

For this programming assignment, feel free to investigate extra datasets and models that you find pertinent and engaging. You may collaborate, but each student is required to turn in their own work. Be sure to reference all external sources properly and acknowledge any contributions.

The objective of this programming assignment is to design, implement, and fine-tune custom deep learning models using the Hugging Face `transformers` library. The project will involve using models from Hugging Face, building custom architectures, and fine-tuning them for the **Flickr30k** dataset. This dataset is widely used for tasks like image captioning, multimodal learning, and visual grounding. The project aims to give students hands-on experience in working with Hugging Face models.

Submit a Jupyter Notebook (.ipynb) with your code, analysis, and visualizations for each problem. Please follow the standard Python programming guidelines. Provide a summary of your findings, challenges faced, and conclusions drawn from each problem.

Tasks

The project will focus on creating and training models for the tasks based on the Flickr30k dataset:

1. Image Captioning: Generate captions based on images from the Flickr30k dataset. Use a subset (training) of the images and captions for training, then use the remaining images as test data to generate captions, compare the generated cation with the origianl captions.
2. Multimodal Learning: Combine textual and visual data to build a hybrid model that can extract key works from the caption to be used as labels.
3. Visual Grounding: Link specific parts of the image to phrases in a caption, e.g., where is the person on the image.

Objective

- Build, fine-tune, and evaluate custom models using Hugging Face's `transformers` library.
- Leverage models suitable for multimodal tasks, such as image captioning and text-vision fusion, and fine-tune them for the Flickr30k dataset.
- Use the Hugging Face developer guide to adapt and extend models.
- Compare the performance of custom models to pre-trained Hugging Face models.

Steps

Step 1: Dataset Preparation

- Load the Flickr30k dataset, which can be accessed through the Hugging Face `datasets` library.
- Explore the dataset, focusing on both images and their associated captions.
- Preprocess the dataset by tokenizing the text captions and preparing the image data for model input (e.g., resizing, normalization).

```
from datasets import load_dataset

# Load the Flickr30k dataset
dataset = load_dataset("flickr30k", split="train")
```

Step 2: Model Selection and Customization

- Choose a pre-trained model from the Hugging Face Model Hub that supports multimodal tasks (e.g., image captioning or text-image retrieval). Suitable models include:
  - `BLIP` (Bootstrapping Language-Image Pre-training)
  - `CLIP` (Contrastive Language-Image Pretraining)
  - `ViLT` (Vision-and-Language Transformer)
- Customize the pre-trained model by fine-tuning it on the Flickr30k dataset, or create a new model class by extending `PreTrainedModel` from the `transformers` library.
- Optionally modify the model architecture to better handle the multimodal nature of the task (e.g., adding attention mechanisms to improve cross-modal learning).

```
from transformers import BlipForConditionalGeneration, BlipProcessor

# Load pre-trained BLIP model
model = BlipForConditionalGeneration.from_pretrained("Salesforce/blip-image-captioning-base")
processor = BlipProcessor.from_pretrained("Salesforce/blip-image-captioning-base")
```

Step 3: Training and Fine-tuning

- Define the training arguments using `TrainingArguments` from the Hugging Face library. Set up parameters like learning rate, batch size, and number of epochs.
- Fine-tune the model on the Flickr30k dataset. The Hugging Face `Trainer` class can be used to simplify the training process.
- Ensure that you evaluate the model using relevant metrics, such as BLEU or ROUGE for image captioning tasks.

```
from transformers import Trainer, TrainingArguments

training_args = TrainingArguments(
    output_dir="./results",
    evaluation_strategy="epoch",
    per_device_train_batch_size=16,
    per_device_eval_batch_size=16,
    learning_rate=5e-5,
    num_train_epochs=3,
    weight_decay=0.01
)

trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=dataset["train"],
    eval_dataset=dataset["test"]
)
```

```
trainer.train()
```

Step 4: Model Evaluation and Comparison

- After training, evaluate the model's performance using relevant evaluation metrics.
- Compare the performance of your custom model with a pre-trained model (e.g., BLIP or CLIP) on the same task.
- Highlight key differences in performance, training efficiency, and ability to generalize.

Canvas Submission

1. Jupyter Notebook: The notebook should include the full implementation of the custom model, data loading, training, evaluation, and comparison with a pre-trained model (if applicable).
2. Project Report (this can be integrated into your Jupyter Notebook): The report should detail:
   - Problem formulation and task selection.
   - Model architecture and design choices.
   - Data preprocessing and augmentation techniques.
   - Model training process and evaluation results.
   - Comparison of custom model performance against pre-trained models.

Grading Criteria

- Data Exploration: 5%
  Demonstrate a clear understanding of the dataset's structure, including:
  - Data types (images, text, labels, etc.)
  - Number of samples (training, validation, test splits)
  - Features/attributes and labels (for supervised tasks)
  - Missing data (if applicable)
  - Imbalances in the dataset (e.g., class distribution)
- Data Preprocessing: 5%
  Ddiscuss the preprocessing and cleaning steps necessary to prepare the dataset for modeling:
  - Handling missing values (if applicable)
  - Normalization, scaling, or standardization (for numerical data)
  - Data augmentation (if applicable for image or text data)
  - Tokenization, padding, and truncation (for text datasets)
  - Any necessary transformations or feature engineering
- Model Design and Implementation: 40%
  Creativity and effectiveness of the model architecture and implementation.
- Training and Fine-tuning: 30%
  Success in fine-tuning the model and achieving reasonable performance on the Flickr30k dataset.

- Evaluation and Comparison: 15%
  Depth of comparison between the custom model and Hugging Face pre-trained models, as well as insights gained.
- Documentation: 5%
  Quality of the report, including clarity, visualizations, and discussion of the results.

Recommended Timeline

- Week of Module 9: Dataset exploration and Data Preprocessing.
- Weeks of Module 10 and 11: Model selection and customization.
- Week of Module 12: Fine-tuning the model on the Flickr30k dataset.
- Week of Module 13: Model evaluation and comparison with pre-trained models.
- Week of Module 14: Finalize report, and prepare documentation, this can be integrated into your Jupyter Notebook.

References

[1] T. Bäck, D. B. Fogel, and Z. Michalewicz, eds. *Handbook of Evolutionary Computation*. CRC Press, 1997.

[2] T. Baltruaitis, C. Ahuja, and L. P. Morency. "Multimodal machine learning: A survey and taxonomy". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2018).

[3] Richard Bellman. "A Markovian decision process". In: *Journal of mathematics and mechanics* (1957), pp. 679–684.

[4] Y. Bengio and Yann Lecun. *Convolutional Networks for Images, Speech, and Time-Series*. https://www.researchgate.net/publication/2453996. Series. 1997.

[5] H. G. Beyer and H. P. Schwefel. "Evolution strategies – A comprehensive introduction". In: *Natural Computing* 1 (1 2002), pp. 3–52.

[6] Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[7] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006. URL: `https://www.microsoft.com/en-us/research/uploads/prod/2006/01/Bishop-Pattern-Recognition-and-Machine-Learning-2006.pdf`.

[8] Leo Breimann et al. "Classification and regression trees". In: *Pacific Grove, Wadsworth* (1984).

[9] Cameron B Browne et al. "A survey of monte carlo tree search methods". In: *IEEE Transactions on Computational Intelligence and AI in games* 4.1 (2012), pp. 1–43.

[10] Guillaume Chaslot et al. "Monte-carlo tree search: A new framework for game ai". In: *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*. Vol. 4. 1. 2008, pp. 216–217.

[11] T. Chen and C. Guestrin. "XGBoost: A scalable tree boosting system". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2016), pp. 785–794.

[12] Marc Peter Deisenrtoh, A. Aldo Faisal, and Cheng Soon Ong. *Mathematics for Machine Learning*. Cambridge University Press, 2020. ISBN: 978-1108455145.

[13] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. Springer, 2015.

[14] A. P. Engelbrecht. *Fundamentals of Computational Swarm Intelligence*. Wiley, 2005.

[15] Vojtech Franc and Vaclav Hlavac. *Statistical Pattern Recognition Toolbox*. `https://cmp.felk.cvut.cz/cmp/software/stprtool/index.html`.

[16] J. H. Friedman. "Greedy Function Approximation: A Gradient Boosting Machine". In: *Annals of Statistics* (2001).

[17] Keinosuke Fukunaga. *Introduction to Statistical Pattern Recognition*. 1st. Academic Press, 1972. ISBN: 0122698509.

[18] Keinosuke Fukunaga. *Introduction to Statistical Pattern Recognition*. 2nd. Academic Press, 1990. ISBN: 0122698517.

[19] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.

[20] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. URL: `https://doi.org/10.1117/12.2664346`.

[21] Noah D Goodman and Andreas Stuhlmüller. *The design and implementation of probabilistic programming languages*. 2014.

[22] N. Hansen and A. Ostermeier. "Completely Derandomized Self-Adaptation in Evolution Strategies". In: *Evolutionary Computation* 9 (1 2001), pp. 159–195.

[23] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. 2nd. Springer, 2009.

[24] H. Hotelling. "Analysis of a complex of statistical variables into principal components". In: *Journal of Educational Psychology* 24 (1933), pp. 417–441.

[25] Ronald A Howard. "Dynamic programming and markov processes." In: (1960).

[26] A. K. Jain, M. N. Murty, and P. J. Flynn. "Data clustering: A review". In: *ACM Computing Surveys* 31 (2 1999), pp. 264–323.

[27] G. James et al. *An Introduction to Statistical Learning*. Springer, 2013.

[28] Gareth James et al. *An introduction to statistical learning*. Vol. 112. Springer, 2013.

[29] J. Kennedy and R. Eberhart. "Particle swarm optimization". In: *Proceedings of ICNN'95 - International Conference on Neural Networks* (1995).

[30] Levente Kocsis and Csaba Szepesvári. "Bandit based monte-carlo planning". In: *European conference on machine learning*. Springer. 2006, pp. 282–293.

[31] M. Kuhn and K. Johnson. *Applied Predictive Modeling*. Springer, 2013.

[32] Wei-Yin Loh. "Classification and regression trees". In: *Wiley interdisciplinary reviews: data mining and knowledge discovery* 1.1 (2011), pp. 14–23.

[33] *Machine Learning at Waikato University*. `https://www.cs.waikato.ac.nz/~ml/index.html`.

[34] M. Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, 1998.

[35] Christoph Molnar. *Interpretable Machine Learning: A Guide For Making Black Box Models Explainable*. Independently published, 2022.

[36] Vincent Müller. "Ethics of Artificial Intelligence and Robotics". In: 2020 (Apr. 2020), pp. 1–31.

[37] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[38] John F Nash Jr. "Equilibrium points in n-person games". In: *Proceedings of the national academy of sciences* 36.1 (1950), pp. 48–49.

[39] J. Ngiam et al. "Multimodal deep learning". In: *In Proceedings of the 28th international conference on machine learning* (2011).

[40] Martin J Osborne and Ariel Rubinstein. *A course in game theory*. MIT press, 1994.

[41] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[42] John R Quinlan et al. "Learning with continuous classes". In: *5th Australian joint conference on artificial intelligence*. Vol. 92. World Scientific. 1992, pp. 343–348.

[43] Lawrence R Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition". In: *Proceedings of the IEEE* 77.2 (1989), pp. 257–286.

[44] Joseph Redmon et al. "You Only Look Once: Unified, Real-Time Object Detection". In: *arXiv* 1506.02640 (2016).

[45] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. ""Why Should I Trust You?" Explaining the Predictions of Any Classifier". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2016), pp. 1135–1144. URL: `https://doi.org/10.1145/2939672.2939778`.

[46] B Myerson Roger et al. "Game theory: analysis of conflict". In: *The President and Fellows of Harvard College, USA* 66 (1991).

[47] Lior Rokach and Oded Maimon. "Top-down induction of decision trees classifiers-a survey". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 35.4 (2005), pp. 476–487.

[48] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. 4th. Prentice Hall, 2020.

[49] Robert E. Schapire. "The Boosting Approach to Machine Learning An Overview". In: 2003. URL: `https://api.semanticscholar.org/CorpusID:221284382`.

[50] David Silver et al. “Mastering the game of Go with deep neural networks and tree search”. In: *nature* 529.7587 (2016), pp. 484–489.

[51] L. Sirovich and M. Kirby. “Low-dimensional procedure for the characterization of human faces”. In: *Journal of the Optical Society of America* 4 (3 1987), pp. 519–524.

[52] Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: An introduction*. MIT Press, 1998.

[53] Robert Tibshirani and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer, 2001.

[54] Matthew A. Turk and Alex P. Pentland. “Eigenfaces for recognition”. In: *Journal of Cognitive Neuroscience* 3 (1 1991), pp. 71–86.

[55] Matthew A. Turk and Alex P. Pentland. “Face recognition using eigenfaces”. In: *Proc. IEEE Conference on Computer Vision and Pattern Recognition* (1991), pp. 586–591.

[56] J. Weston, S. Bengio, and N. Usunier. “WSABIE: Scaling up to large vocabulary image annotation”. In: *IJCAI International Joint Conference on Artificial Intelligence* (2011).

[57] Ian H. Witten, Eibe Frank, and Mark A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques*. 3rd. Morgan Kaufmann, 2011.

[58] Shukang Yin et al. *A Survey on Multimodal Large Language Models*. 2024. arXiv: 2306.13549 [cs.CV]. URL: https://arxiv.org/abs/2306.13549.