Department

STEM Engineering Department

Project Title

Multilingual AI Translation: Designing a machine learning model to process multilanguage environments into a defined language field.

Proposal:

Objective Statement:
This project aims to develop a complex algorithm on specified hardware to train an AI model capable of processing multilingual environments into a desired language without relying on cloud architecture. The algorithm will utilize PyTorch library and Python programming language. Python's flexibility will allow the learning model trainer to run on various hardware platforms, avoiding funding bottlenecks on hardware. Once the language model is complete, it will be extended to a Java interface library to ensure compatibility and security across different devices.

Background and Significance:
The goal of this project is to develop a low power language model that can be utilized on a wide range of devices for a wide range of scenarios. These devices might include handheld transcribers, office multilingual devices, and smartphones. This wide range of integration is key to ensuring that the language model can be widely adopted and improves communication across various contexts. When running the language model, it can be implemented in an application to function either standalone or in the background of other applications, enhancing their language content outreach for their user base. This dual functionality will make it a highly flexible tool that can be implemented in multiple different environments.

To build this language model, we will use the M3P data structure as the base for the language trainer. The M3P data structure provides a robust starting point that can be improved upon with our own language training content feeds. This approach allows us to leverage existing technology while customizing and optimizing it to fit our specific needs and goals. Furthermore, we will look into Microsoft's multilingual-model-transfer repository on the new formulated algorithm for cross-lingual transfer learning. This repository presents a potential new method for training language models that may allow us to use lower hardware resources. By exploring this, we aim to make our language model not only effective but also efficient, which is crucial for deployment on low-power devices.

For hardware testing of the language model, we will utilize a Raspberry Pi 5 equipped with an AI HAT+ to process the internal model program on the NPU processor. By choosing the Raspberry Pi for testing allows us to see how our language model handles on a low power device with limited resource capabilities. This will help us understand the necessary improvements for running the model on low power devices or in the background of other applications. Testing on the Raspberry Pi will simulate real-world conditions, ensuring that our language model is both practical and scalable.

Methods with Timeline layout:

January 6th - January 15th:

Build a beta model trainer based off of Microsoft's new CLTL algorithm. This involves understanding the algorithm's structure, setting up the necessary development environment, and coding the initial version of the model trainer.

January 16th - January 29th:

Run extensive tests to determine the best training data structure for achieving a lower loss value on the language model. Various data preprocessing techniques and data augmentation methods will be applied to enhance the training process.

January 30th - February 1st:

Build a Raspberry Pi with AI HAT+ module to run the application for testing the language model. This includes assembling the hardware components, installing the necessary software and drivers, and configuring the system for optimal performance.

February 2nd - February 8th:

Develop a language model application for running tests. This step involves designing and implementing a user-friendly interface, integrating the model with the application, and performing initial testing to ensure functionality.

February 9th - February 15th:

Create an API for device to model integration. The API will allow seamless communication between the language model and various devices, ensuring compatibility and ease of use. Security features will be incorporated to protect data integrity and privacy.

February 16th - March 15th:

Complete the training of the language model. This period will involve multiple iterations of training, fine-tuning hyperparameters, and monitoring performance metrics to achieve the desired accuracy and efficiency.

March 2nd - March 15th:

Alongside training the language model, create a library interface for the model attached to the API. This library will facilitate easy integration of the language model into various applications or devices, providing developers with a flexible and robust toolset.

March 16th - April 12th:

Conduct additional refinements and testing to ensure the model operates efficiently on the Raspberry Pi and other target devices. Performance testing and stress testing will be gathered to make final adjustments.

April 13th - April 29th:

Integrate the language model into various applications or devices and finalize project documentation. Detailed user guides, API documentation, and training materials will be created to support adoption and usage of the model.

Expected Outcome:

The expected outcome of this project is to develop a low-cost, low-power, and open-source language model that is versatile enough to be used across a wide range of devices, breaking down language barriers in various environmental scenarios. The vision is to create a model that enhances communication by providing real-time translations and transcriptions, allowing seamless interaction between people of different linguistic backgrounds.

One of the key achievements will be constructing a working beta model trainer using Microsoft's new CLTL algorithm. This step will ensure that we have a robust foundation upon which to build and improve our language model. By implementing the M3P data structure as a base and incorporating our custom language training content feeds, we aim to enhance the model's accuracy and efficiency. Additionally, we will explore Microsoft's multilingual-model-transfer repository to potentially lower hardware resource requirements, making the model more accessible through open-source APIs.

Rigorous testing will be conducted to identify the best training data structures, aiming for the lowest possible loss value in the language model. This process will include extensive data preprocessing and augmentation techniques to ensure the model performs optimally in diverse scenarios. Building a language model application for running tests on a Raspberry Pi 5, equipped with an AI HAT+, will allow us to assess the model's performance on low-power hardware. This will provide valuable insights into necessary improvements for running the model on various devices, from handheld transcribers to smartphones while allowing others to utilize our findings and create a DIY unit with relatively low cost.

Creating an API for device to model integration will be another significant milestone. This API will facilitate seamless communication between the language model and different devices, ensuring compatibility and ease of use. A library interface will be constructed to enable integration of the model into various applications or devices, providing developers with a flexible and robust toolset.

Ultimately, the successful completion of this project will result in a versatile, efficient language model capable of breaking down language barriers and enabling meaningful interactions in real-time. This will empower people to utilize a pre-built model onto their own devices or utilizing a low cost-built device for translating language content. This will also allow high level individuals to construct their own system using the open-source API and DIY documentation.

Literature Review:

M. Virvou, C. Troussas and E. Alepis, "Machine learning for user modeling in a multilingual learning system," *International Conference on Information Society (i-Society 2012)*, London, UK, 2012, pp. 292-297.

D. Sculley and C. E. Brodley, "Compression and machine learning: a new perspective on feature space vectors," *Data Compression Conference (DCC'06)*, Snowbird, UT, USA, 2006, pp. 332-341, doi: 10.1109/DCC.2006.13.

Mohammed, M., Khan, M.B., & Bashier, E.B.M. (2016). Machine Learning: Algorithms and Applications (1st ed.). CRC Press. https://doi.org/10.1201/9781315371658

Bonaccorso, Giuseppe. Machine Learning Algorithms: Popular Algorithms for Data Science and Machine Learning, 2nd Edition. Germany, Packt Publishing, 2018.

Hao Li, Gopi Krishnan Rajbahadur, and Cor-Paul Bezemer. 2024. Studying the Impact of TensorFlow and PyTorch Bindings on Machine Learning Software Quality. ACM Trans. Softw. Eng. Methodol. Just Accepted (July 2024). https://doi.org/10.1145/3678168

Work and Experience:

I have experience in creating machine learning models using PyTorch as my framework library. The work I completed was built on hardware sourced from my own resources, using a GTX 1080 Ti for its CUDA cores. This model was trained to remove backgrounds from images with 95% accuracy. Through this project, I successfully utilized the model to remove over 35,000 backgrounds from images for a furniture store called Havertys.

In addition to this, my project involved extensive data preprocessing and augmentation to ensure high-quality input data, which contributed to the model's impressive accuracy. I also developed scripts for automating the image processing tasks, which significantly improved efficiency and productivity. The success of this project not only demonstrates my technical skills in machine learning and deep learning but also my ability to manage and execute projects independently using my own hardware and resources. This experience has equipped me with practical skills and confidence to tackle this project for a machine learning multilanguage model. For more information on the work experience project review my breakdown here https://techdbernstein.com/portfolio/ai-background-remover/.

IRB/IACUC:

No. This multilanguage model will not require the need for any IRB or IACUC. This project will be created through a machine learning algorithm that is created by referencing the CLTL structure.

Budget:

The budget will be around $1,200-$1,500. Most of the budget will be spent on acquiring a high VRAM and CUDA core GPU unit. The GPU units that will be researched are A900 and RTX units. The rest of the cost will be used to purchase Raspberry Pis for each developer that needs to run the code.