Connor Gag

Machine Learning Project Portfolio

gagconnor@gmail.com cgag@ucsd.com +1-507-766-6008
github.com/connorgag connorgag.github.io linkedin.com/in/connorgag

Overview

I am a Machine Learning Engineer with expertise in AI, NLP, and computer vision. My work focuses on developing intelligent systems that enhance decision-making and automate complex tasks. This portfolio showcases my key projects in deep learning, natural language processing, and computer vision, demonstrating my technical skills and problem-solving abilities.

Featured Projects

Custom Fine-Tuning and Contrastive Learning with Transformers

GitHub: Custom-Fine-Tuning-and-Contrastive-Learning

- Implemented and evaluated multiple techniques for fine-tuning BERT on scenario classification tasks, including Stochastic Weight Averaging (SWA), Top Layer Reinitialization, Low-Rank Adaptation (LoRA), and Contrastive Learning.
- Achieved 91.15% validation accuracy by combining SWA and Top Layer Reinitialization techniques, outperforming the baseline model's 89.61% accuracy.
- Conducted comparative analysis of supervised (SupCon) and self-supervised (SimCLR) contrastive learning approaches, demonstrating superior performance of supervised methods (90.41% vs 81.81% accuracy).
- Experimented with LoRA at different ranks (8, 16, 32) to reduce trainable parameters to less than 1% while maintaining reasonable performance, decreasing per-epoch training time by approximately 50%.
- Analyzed the impact of reinitialization depth on model performance, finding optimal results when reinitializing the top 2 layers of the pretrained BERT encoder.

Transformer Encoder and Decoder Optimization

GitHub: Low-Level-Transformer-Encoder-and-Decoder-Optimization

- Implemented and analyzed attention mechanisms in a transformer, optimizing encoder and decoder attention patterns
- Evaluated different positional embedding strategies (learned, sinusoidal, and ALiBi), reducing perplexity from 178.20 to 112.86
- Improved classification accuracy from 33.6% to 85.86% and conducted architecture exploration to enhance model efficiency

Analyzing Changing Trends of Podcasts Using BERTopic

GitHub: BERTopic-Podcast-Topic-Modeling

- Applied BERTopic to extract and analyze key topics from podcast transcripts over time, optimizing for both structured and conversational audio
- Processed and segmented over 3,000 YouTube transcripts, handling missing timestamps and reducing conversational noise for improved topic modeling
- Enhanced topic labels using GPT-4o and conducted time-series analysis to track evolving trends in podcasts
- Achieved a classification rate of 78.56% for news podcasts and 37.98% for conversational podcasts, demonstrating the model's effectiveness in identifying key topics from noisy data

RNN and LSTM Shakespearean Text Generation

GitHub: RNN-and-LSTM-Shakespearean-Text-Generation

- Implemented and compared Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) models for character-level text generation using Shakespeare's works
- Experimented with varying sequence lengths, hidden layer sizes, and teacher forcing to optimize model performance, achieving a validation loss of 1.36 with enhanced LSTM architectures
- Generated synthetic Shakespearean text using temperature-controlled sampling, demonstrating the trade-off between coherence and creativity in AI-generated content

Semantic Segmentation of Images Using Deep Learning

GitHub: Semantic-Segmentation-of-Images

- Developed multiple CNN architectures for semantic segmentation on the PASCAL VOC-2012 dataset, including a baseline FCN, custom architecture, ResNet-34 based model, and U-Net implementation
- Implemented various optimization techniques including cosine annealing learning rate scheduling, data augmentation (random cropping, flipping), and weighted loss functions to address class imbalance
- Achieved significant performance improvements with the pretrained ResNet-based model, reaching 83.3% pixel accuracy and 0.188 mean IoU, compared to baseline accuracy of 73.77% and IoU of 0.0665
- Experimented with different neural network architectures and compared their performance, including a custom network with strategic dropout layers and a U-Net implementation with batch normalization
- Conducted comprehensive analysis of model performance using pixel accuracy and Intersection-over-Union (IoU) metrics, with detailed visualization of results and loss curves

Predictive Health Assessment Model

GitHub: Predictive-Health-Assessment-Model

- Used Azure Machine Learning Studio to train and hyperparameter-tune a model to predict individuals' general health given survey data
- Deployed as a scalable API using Azure, enabling quick and secure health assessments via internal endpoints

Image Classification with Low-Level Neural Network

GitHub: Classifying-Fashion-MNIST

- Implemented a neural network from scratch using NumPy to classify pictures of clothing without relying on high-level machine learning libraries like TensorFlow or PyTorch
- Experimented with different activation functions, regularization techniques, and hyperparameter tuning to improve model performance, achieving a test accuracy of 88%
- Analyzed the effects of L1 vs. L2 regularization and momentum-based optimization on convergence and accuracy
- Visualized and preprocessed dataset by normalizing pixel values and splitting data into training, validation, and test sets

Expectation Maximization on Movie Reviews

GitHub: Expectation-Maximization-on-Movie-Reviews

- Classified individuals into 4 types of movie watchers to predict each person's future reviews
- Accomplished this through applying 256 iterations of Expectation Maximization on a dataset of movie reviews

N-Gram Log Likelihood

GitHub: N-Gram-Log-Likelihood

- Built a program that computes the unigram and bigram log likelihood of a sequence of characters, tokenizing by word
- Combined the unigram and bigram models to compute the log likelihood of a sentence

Technical Skills

- **Deep Learning:** Transformers, CNNs, RNNs, LSTMs, Attention Mechanisms
- **NLP:** BERTopic, Text Generation, Topic Modeling, NLTK, Hugging Face, RAG
- **Computer Vision:** Semantic Segmentation, Image Classification, Object Detection
- **Languages:** Python, SQL, Java, R, Bash
- **Python Libraries:** TensorFlow, PyTorch, Pandas, NumPy, Scikit-learn, Matplotlib, PySpark, Selenium
- **Cloud & DevOps:** Google Cloud Platform (GCP), Azure, Docker, Snowflake
- **Databases:** PostgreSQL, OracleDB
- **Tools:** Git, Jupyter, Anaconda, PowerBI, Unity, Jira
- **Methods:** REST APIs, Linux command line, Expectation Maximization, Hyperparameter Tuning

For more details and additional projects, please visit:

github.com/connorgag connorgag.github.io linkedin.com/in/connorgag