Neural Network “MENU” for projects

Neural Networks Inception and Their Variations

Project 1: Implementation and Analysis of Inception-v4

1. **Title of the Project**: Implementation and Analysis of Inception-v4 for Image Classification
2. **Literature Overview**: Inception-v4 is an advanced version of Inception architecture that integrates residual connections to improve training efficiency and accuracy. For a comprehensive understanding, refer to the paper "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning" by Szegedy et al.
3. **Source of the Code**: The official TensorFlow models repository provides the implementation of Inception-v4. Access it here: TensorFlow Models
4. **Choice of the Platform**: Google Colab offers a free and convenient platform for implementing and experimenting with Inception-v4. Access Colab here: Google Colab
5. **Dataset Management and Acquisition**: Utilize the CIFAR-100 dataset, which contains 100 classes of images. Divide the dataset into 70% training, 15% validation, and 15% testing sets to ensure robust model evaluation.
6. **Analytics of the Source Code**: Develop a flow diagram illustrating the data preprocessing steps, the architecture of Inception-v4, and the training process. This will demonstrate a clear understanding of the code and model workflow.
7. **Results Analysis**: Evaluate the model's performance using accuracy metrics and present a confusion matrix to analyze classification errors across different classes.
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

Project 2: Comparative Study of Inception-v3 and Inception-ResNet-v2

1. **Title of the Project**: Comparative Study of Inception-v3 and Inception-ResNet-v2 in Image Recognition Tasks
2. **Literature Overview**: Inception-v3 and Inception-ResNet-v2 are significant advancements in the Inception series, with the latter incorporating residual connections. For detailed insights, refer to "Rethinking the Inception Architecture for Computer Vision" and "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning" by Szegedy et al.
3. **Source of the Code**: Implementations for both models are available in the Keras library. Access them here: Keras Applications
4. **Choice of the Platform**: Use Jupyter Notebooks within the Anaconda environment for implementation. Download Anaconda here: Anaconda Distribution

5. **Dataset Management and Acquisition**: Employ the ImageNet subset dataset, focusing on a selected number of classes due to computational constraints. Split the data into 60% training, 20% validation, and 20% testing sets.
6. **Analytics of the Source Code**: Create flow diagrams for both models, highlighting their architectural differences and data flow during training and inference.
7. **Results Analysis**: Compare the performance of both models using metrics such as accuracy, precision, recall, and F1-score. Present findings in tabular and graphical formats.
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

Project 3: Transfer Learning with Inception-v3 for Medical Image Classification

1. **Title of the Project**: Transfer Learning with Inception-v3 for Classifying Medical Images
2. **Literature Overview**: Inception-v3 has been effectively utilized in various image classification tasks, including medical imaging. For an in-depth understanding, refer to "Rethinking the Inception Architecture for Computer Vision" by Szegedy et al.
3. **Source of the Code**: The Keras library provides a pre-trained Inception-v3 model suitable for transfer learning. Access it here: Keras Applications
4. **Choice of the Platform**: Google Colab is recommended for its GPU support, facilitating efficient training. Access Colab here: Google Colab
5. **Dataset Management and Acquisition**: Utilize a publicly available medical image dataset, such as the Chest X-Ray Images dataset. Divide the data into 70% training, 15% validation, and 15% testing sets.
6. **Analytics of the Source Code**: Illustrate the transfer learning process with a flow diagram, detailing the steps of feature extraction, model fine-tuning, and classification.
7. **Results Analysis**: Assess the model's performance using metrics like accuracy, sensitivity, specificity, and AUC-ROC curves.
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

ResNet Networks and Their Variations

Project 4: Exploring ResNet-50 for Image Classification

1. **Title of the Project**: Exploring ResNet-50 Architecture for Image Classification Tasks
2. **Literature Overview**: ResNet-50 is a widely used deep residual network known for its effectiveness in image classification. For more details, refer to the paper "Deep Residual Learning for Image Recognition".
3. **Source of the Code**: Official PyTorch implementation: ResNet in PyTorch
4. **Choice of the Platform**: Google Colab with GPU acceleration. Access Colab here: Google Colab
5. **Dataset Management and Acquisition**: Use the CIFAR-10 dataset, splitting it into 70% training, 15% validation, and 15% testing.
6. **Analytics of the Source Code**: Create a flow diagram detailing ResNet-50's architecture, residual blocks, and data flow.
7. **Results Analysis**: Evaluate performance using a confusion matrix, precision, recall, and F1-score.
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

Project 5: ResNet-152 vs. ResNet-101 - A Comparative Study

1. **Title of the Project**: Comparative Study of ResNet-152 and ResNet-101 for Image Recognition
2. **Literature Overview**: ResNet-101 and ResNet-152 improve feature extraction via deeper architectures. For more details, refer to the paper "Identity Mappings in Deep Residual Networks".
3. **Source of the Code**: PyTorch provides implementations: ResNet Models
4. **Choice of the Platform**: VSCode with PyTorch. Download VSCode here: Visual Studio Code
5. **Dataset Management and Acquisition**: ImageNet subset with 60% training, 20% validation, and 20% testing.
6. **Analytics of the Source Code**: Develop flowcharts highlighting architectural differences.
7. **Results Analysis**: Compare accuracy, computational efficiency, and model size.
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is

recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

Project 6: Transfer Learning with ResNet-50 for Medical Image Segmentation

1. **Title of the Project**: Transfer Learning Using ResNet-50 for Lung Disease Detection
2. **Literature Overview**: ResNet-50 is frequently used for transfer learning in medical imaging. For more details, refer to the paper "Transfer Learning in Medical Imaging".
3. **Source of the Code**: Pretrained ResNet-50 in Keras. Access it here: Keras Applications
4. **Choice of the Platform**: Hugging Face Spaces for easy deployment. Access Hugging Face here: Hugging Face
5. **Dataset Management and Acquisition**: Chest X-Ray dataset with 70% training, 15% validation, and 15% testing.
6. **Analytics of the Source Code**: Visualize feature extraction layers in a flowchart.
7. **Results Analysis**: Use AUC-ROC curves and sensitivity/specificity for evaluation.
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

YOLO Networks and Their Variations

Project 7: Object Detection with YOLOv8

1. **Title of the Project**: Real-Time Object Detection with YOLOv8
2. **Literature Overview**: YOLOv8 improves detection speed and accuracy. For more details, refer to the YOLOv8 paper.
3. **Source of the Code**: Official implementation: YOLOv8 GitHub
4. **Choice of the Platform**: Google Colab for training and inference. Access Colab here: Google Colab
5. **Dataset Management and Acquisition**: Use the COCO dataset with 70% training, 15% validation, and 15% testing.
6. **Analytics of the Source Code**: Provide a flow diagram of YOLOv8's architecture.
7. **Results Analysis**: Evaluate precision, recall, and mean Average Precision (mAP).

8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

Project 8: Comparing YOLOv7 and YOLOv8 for Traffic Surveillance

1. **Title of the Project**: YOLOv7 vs. YOLOv8: Detecting Traffic Violations
2. **Literature Overview**: YOLOv7 introduced optimizations in object detection. For more details, refer to the YOLOv7 paper.
3. **Source of the Code**: YOLOv7 GitHub
4. **Choice of the Platform**: VSCode with PyTorch integration. Download VSCode here: Visual Studio Code
5. **Dataset Management and Acquisition**: Traffic surveillance dataset, split into 70% training, 15% validation, and 15% testing.
6. **Analytics of the Source Code**: Compare architectural changes between YOLOv7 and YOLOv8.
7. **Results Analysis**: Evaluate model accuracy and inference speed.
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

Segment Anything Models and Their Variations

Project 10: Implementing Meta's Segment Anything Model (SAM)

1. **Title of the Project**: Segmentation with Meta's Segment Anything Model (SAM)
2. **Literature Overview**: SAM is a powerful segmentation model by Meta AI. For more details, refer to the paper "Segment Anything".
3. **Source of the Code**: Official GitHub Repo: Segment Anything
4. **Choice of the Platform**: Hugging Face Spaces for web deployment. Access Hugging Face here: Hugging Face
5. **Dataset Management and Acquisition**: Use the ADE20K dataset.
6. **Analytics of the Source Code**: Illustrate the model pipeline using a flowchart.
7. **Results Analysis**: Evaluate segmentation accuracy using IoU metrics.

8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

Project 11: Comparing SAM with U-Net for Medical Image Segmentation

1. **Title of the Project**: Medical Image Segmentation: SAM vs. U-Net
2. **Literature Overview**: U-Net is a classical segmentation model in medical imaging. For more details, refer to the paper "U-Net: Convolutional Networks for Biomedical Image Segmentation".
3. **Source of the Code**: U-Net GitHub
4. **Choice of the Platform**: Google Colab with PyTorch. Access Colab here: Google Colab
5. **Dataset Management and Acquisition**: Use a medical segmentation dataset.
6. **Analytics of the Source Code**: Compare architectures using a flowchart.
7. **Results Analysis**: Compare Dice Score and IoU metrics.
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

Project 12: Fine-Tuning SAM for Satellite Image Segmentation

1. **Title of the Project**: Enhancing Satellite Image Segmentation with SAM
2. **Literature Overview**: SAM can be fine-tuned for satellite imagery. For more details, refer to the paper "Segment Anything".
3. **Source of the Code**: Segment Anything Repository
4. **Choice of the Platform**: Hugging Face Spaces. Access Hugging Face here: Hugging Face
5. **Dataset Management and Acquisition**: Remote sensing dataset (DeepGlobe).
6. **Analytics of the Source Code**: Show workflow in a flowchart.
7. **Results Analysis**: Evaluate IoU and pixel accuracy.
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

GAN Neural Networks and Their Variations

Project 13: Exploring StyleGAN3 for High-Fidelity Image Generation

1. **Title of the Project**: High-Fidelity Image Generation Using StyleGAN3
2. **Literature Overview**: StyleGAN3 addresses aliasing issues present in previous versions, resulting in improved image quality and consistency. For more details, refer to the paper "Alias-Free Generative Adversarial Networks".
3. **Source of the Code**: The official implementation of StyleGAN3 is available on GitHub: StyleGAN3 Repository
4. **Choice of the Platform**: Google Colab provides a suitable environment with GPU support for training StyleGAN3 models. Access Colab here: Google Colab
5. **Dataset Management and Acquisition**: Utilize the FFHQ (Flickr-Faces-HQ) dataset, which contains high-quality images of human faces. Split the dataset into 70% training, 15% validation, and 15% testing sets to ensure comprehensive evaluation.
6. **Analytics of the Source Code**: Develop a flow diagram illustrating the architecture of StyleGAN3, highlighting the generator, discriminator, and the alias-free mechanisms implemented. This will demonstrate a clear understanding of the model's structure and data flow.
7. **Results Analysis**: Evaluate the generated images using metrics such as Fréchet Inception Distance (FID) to assess image quality and diversity.
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

Project 14: Implementing CycleGAN for Unpaired Image-to-Image Translation

1. **Title of the Project**: Unpaired Image-to-Image Translation Using CycleGAN
2. **Literature Overview**: CycleGAN enables image translation between two domains without paired examples, making it effective for tasks like style transfer. For more details, refer to the paper "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks".
3. **Source of the Code**: The official PyTorch implementation of CycleGAN is available here: CycleGAN and pix2pix in PyTorch
4. **Choice of the Platform**: VSCode with the PyTorch framework is suitable for implementing CycleGAN. Download VSCode here: Visual Studio Code

5. **Dataset Management and Acquisition**: Use the Yosemite Summer to Winter dataset, which contains unpaired images of Yosemite in different seasons. Divide the dataset into 80% training, 10% validation, and 10% testing sets.
6. **Analytics of the Source Code**: Create a flow diagram depicting the CycleGAN architecture, including the two generators and two discriminators, as well as the cycle consistency loss mechanism.
7. **Results Analysis**: Assess the quality of translated images using qualitative visual comparisons and quantitative metrics like FID.
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

Project 15: Enhancing Image Resolution with ESRGAN

1. **Title of the Project**: Image Super-Resolution Using Enhanced Super-Resolution GAN (ESRGAN)
2. **Literature Overview**: ESRGAN improves upon traditional GAN-based super-resolution methods by introducing a Residual-in-Residual Dense Block, leading to better performance. For more details, refer to the paper "ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks".
3. **Source of the Code**: The ESRGAN implementation can be found here: ESRGAN Repository
4. **Choice of the Platform**: Google Colab is recommended for its GPU capabilities, facilitating efficient training of the ESRGAN model. Access Colab here: Google Colab
5. **Dataset Management and Acquisition**: Employ the DIV2K dataset, which is commonly used for super-resolution tasks. Split the dataset into 70% training, 15% validation, and 15% testing sets.
6. **Analytics of the Source Code**: Illustrate the ESRGAN architecture through a flow diagram, focusing on the generator's Residual-in-Residual Dense Blocks and the discriminator's structure.
7. **Results Analysis**: Evaluate the super-resolved images using Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) metrics.
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

U-Net Neural Networks and Their Variations

Project 16: Attention U-Net for Improved Medical Image Segmentation

1. **Title of the Project**: Enhancing Medical Image Segmentation with Attention U-Net
2. **Literature Overview**: Attention U-Net introduces attention mechanisms to improve focus on relevant image regions, leading to better segmentation results. For more details, refer to the paper "Attention U-Net: Learning Where to Look for the Pancreas".
3. **Source of the Code**: Official PyTorch implementation available here: Attention U-Net GitHub
4. **Choice of the Platform**: Google Colab for cloud-based training. Access Colab here: Google Colab
5. **Dataset Management and Acquisition**: Use the CHAOS medical dataset (Computed Tomography images for organ segmentation). Split into 70% training, 15% validation, and 15% testing.
6. **Analytics of the Source Code**: Develop a flow diagram showcasing the Attention U-Net's encoder-decoder structure and the attention gate mechanism.
7. **Results Analysis**: Use Dice Coefficient, Intersection over Union (IoU), and precision-recall metrics for evaluation.
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

Project 17: Implementing 3D U-Net for Volumetric Medical Image Segmentation

1. **Title of the Project**: 3D U-Net for Segmentation of Volumetric Medical Images
2. **Literature Overview**: 3D U-Net extends the original U-Net architecture to three dimensions, making it suitable for segmenting volumetric medical data such as CT and MRI scans. For more details, refer to the paper "3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation".
3. **Source of the Code**: Official 3D U-Net PyTorch implementation: 3D U-Net GitHub
4. **Choice of the Platform**: Google Colab with GPU acceleration for model training and inference. Access Colab here: Google Colab. Hugging Face Spaces for model deployment and interactive segmentation demos. Access Hugging Face here: Hugging Face
5. **Dataset Management and Acquisition**: Dataset: BraTS (Brain Tumor Segmentation Dataset). This dataset provides multi-modal MRI scans for brain tumor segmentation. Data Split: 70% training, 15% validation, 15% testing to ensure balanced evaluation across different tumor types.

6. **Analytics of the Source Code**: Illustrate the 3D U-Net pipeline using a flow diagram, covering preprocessing (resampling, normalization), 3D convolutional encoder-decoder structure, and skip connections and upsampling strategies.
7. **Results Analysis**: Evaluate model performance using Dice Coefficient, IoU (Intersection over Union), and Hausdorff Distance for segmentation accuracy. Compare 3D U-Net with 2D U-Net to assess the benefits of 3D spatial modeling.
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

Project 18: Swin-Unet for Transformer-Based Medical Image Segmentation

1. **Title of the Project**: Using Swin-Unet for Efficient Medical Image Segmentation
2. **Literature Overview**: Swin-Unet is a transformer-based variation of U-Net that leverages Swin Transformers for global feature learning, achieving state-of-the-art results in medical imaging. For more details, refer to the paper "Swin-Unet: Unet-like Pure Transformer for Medical Image Segmentation".
3. **Source of the Code**: Official Swin-Unet implementation in PyTorch: Swin-Unet GitHub
4. **Choice of the Platform**: Hugging Face Spaces for cloud-based inference. Access Hugging Face here: Hugging Face
5. **Dataset Management and Acquisition**: Use the ISIC Skin Cancer Segmentation dataset. Divide into 75% training, 15% validation, and 10% testing.
6. **Analytics of the Source Code**: Illustrate the Swin Transformer block integration into U-Net's encoder-decoder framework through a flow diagram.
7. **Results Analysis**: Compare IoU, Dice Score, and segmentation accuracy with traditional U-Net models.
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

Vision Transformers (ViTs)

Project 19: Implementing Data-efficient Image Transformers (DeiT) for Image Classification

1. **Title of the Project**: Data-efficient Image Transformers (DeiT) for Image Classification
2. **Literature Overview**: DeiT is a Vision Transformer variant that enhances training efficiency and performance without requiring extensive datasets. It introduces a teacher-student distillation approach to improve accuracy. For more details, refer to the paper "Training Data-Efficient Image Transformers & Distillation Through Attention".
3. **Source of the Code**: The official implementation of DeiT is available on GitHub: DeiT GitHub Repository
4. **Choice of the Platform**: Google Colab provides a suitable environment with GPU support for training DeiT models. Access Colab here: Google Colab
5. **Dataset Management and Acquisition**: Utilize the CIFAR-100 dataset, which contains 100 classes of images. Split the dataset into 70% training, 15% validation, and 15% testing sets to ensure robust model evaluation.
6. **Analytics of the Source Code**: Develop a flow diagram illustrating the DeiT architecture, highlighting the distillation token mechanism and the training process. This will demonstrate a clear understanding of the model's structure and data flow.
7. **Results Analysis**: Evaluate the model's performance using accuracy metrics and present a confusion matrix to analyze classification errors across different classes.
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

Project 20: Exploring Swin Transformer for Object Detection

1. **Title of the Project**: Object Detection Using Swin Transformer
2. **Literature Overview**: Swin Transformer is a hierarchical Vision Transformer that computes representations with shifted windows, achieving linear computational complexity and improved performance in object detection tasks. For more details, refer to the paper "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows".
3. **Source of the Code**: The official implementation of Swin Transformer is available on GitHub: Swin Transformer GitHub Repository
4. **Choice of the Platform**: VSCode with the PyTorch framework is suitable for implementing Swin Transformer. Download VSCode here: Visual Studio Code

5. **Dataset Management and Acquisition**: Use the COCO dataset, which is widely used for object detection tasks. Divide the dataset into 80% training, 10% validation, and 10% testing sets.
6. **Analytics of the Source Code**: Create a flow diagram depicting the Swin Transformer architecture, including the shifted window mechanism and hierarchical structure.
7. **Results Analysis**: Assess the model's performance using metrics such as mean Average Precision (mAP) and Intersection over Union (IoU).
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

Project 21: Applying Pyramid Vision Transformer (PVTv2) for Semantic Segmentation

1. **Title of the Project**: Semantic Segmentation with Pyramid Vision Transformer v2 (PVTv2)
2. **Literature Overview**: PVTv2 improves upon the original PVT by introducing features like linear complexity attention layers and overlapping patch embedding, enhancing performance in semantic segmentation tasks. For more details, refer to the paper "Pyramid Vision Transformer v2: More Efficient and Accurate Visual Recognition with Transformers".
3. **Source of the Code**: The official implementation of PVTv2 is available on GitHub: PVTv2 GitHub Repository
4. **Choice of the Platform**: Google Colab with GPU support is recommended for training PVTv2 models. Access Colab here: Google Colab
5. **Dataset Management and Acquisition**: Employ the ADE20K dataset, which is commonly used for semantic segmentation tasks. Split the dataset into 70% training, 15% validation, and 15% testing sets.
6. **Analytics of the Source Code**: Illustrate the PVTv2 architecture through a flow diagram, focusing on the pyramid structure and attention mechanisms.
7. **Results Analysis**: Evaluate the model's performance using metrics such as Pixel Accuracy and Mean Intersection over Union (mIoU).
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

Project Proposals for Local Conversational Agents vs. Cloud-Based LLM Comparison

These projects focus on setting up and testing local LLM models and comparing their performance with a cloud-based model using objective evaluation criteria.

Project 22: Evaluating Local DeepSeek LLM vs. Cloud-Based Conversational Agent

1. **Title of the Project**: Comparing Local DeepSeek LLM Model with a Cloud-Based LLM for Conversational AI
2. **Literature Overview**: DeepSeek LLM is an open-source model optimized for local deployment. For more details, refer to the paper "DeepSeek LLM - Open-Source AI Models".
3. **Source of the Code**: Official DeepSeek model repository: DeepSeek GitHub
4. **Choice of the Platform**: Local Setup: Run DeepSeek on a high-performance local machine with GPU acceleration using Ollama. Cloud Model for Comparison: Use OpenAI GPT-4 or Claude AI via an API.
5. **Dataset Management and Acquisition**: Create a benchmark dataset for conversational queries, including general knowledge (e.g., history, science), coding assistance (e.g., Python debugging), and creative tasks (e.g., story writing). Evaluate responses using this dataset.
6. **Analytics of the Source Code**: Flow diagram of DeepSeek LLM inference pipeline vs. Cloud model inference. Memory consumption, latency comparison, token generation efficiency.
7. **Results Analysis**: Define comparison metrics: Response Latency (ms), Accuracy (BLEU score, ROUGE for text comparison), Computational Cost (GPU usage, RAM consumption). Generate graphs comparing these metrics.
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

Project 23: Comparing Local Qwen LLM vs. Cloud-Based LLM Performance

1. **Title of the Project**: Benchmarking Local Qwen LLM Against a Cloud-Based Conversational AI Model
2. **Literature Overview**: Qwen LLM is an open-source language model by Alibaba Cloud, designed for efficient local execution. For more details, refer to the paper "Qwen LLM - A Versatile Language Model".
3. **Source of the Code**: Qwen GitHub

4. **Choice of the Platform**: Local Model Deployment: Run Qwen on a NVIDIA RTX 4090+ GPU setup via Ollama. Cloud-Based Alternative: Google Gemini API or Claude AI API.
5. **Dataset Management and Acquisition**: Same benchmark dataset as Project 22, but include translation tasks (e.g., English to Chinese, French) and mathematical reasoning (word problem solving).
6. **Analytics of the Source Code**: Flow diagram showing Qwen's attention mechanism, token processing pipeline, and local memory usage. Compare model's speed and inference time on different hardware configurations.
7. **Results Analysis**: Define comparison metrics: Translation Accuracy (BLEU score), Mathematical Reasoning Accuracy (%), Inference Speed (tokens/sec). Graph-based comparison of local Qwen vs. cloud LLM performance.
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

Project 24: Testing Local Llama 3 LLM vs. a Cloud-Based LLM

1. **Title of the Project**: How Well Does Llama 3 Perform Locally Compared to Cloud-Based LLMs?
2. **Literature Overview**: Llama 3 is a high-performing open-source LLM optimized for on-device execution. Meta's research on efficient model scaling allows Llama to compete with cloud-hosted models. For more details, refer to the paper "Introducing Llama 3: Open-Source AI".
3. **Source of the Code**: Llama 3 GitHub
4. **Choice of the Platform**: Local Deployment: Run Llama 3 on Ollama with optimized quantized versions. Cloud-Based Comparison: Compare with Anthropic Claude or GPT-4 API.
5. **Dataset Management and Acquisition**: Expand dataset with code generation tasks (JavaScript, Rust) and logic-based reasoning (Chess strategy, Sudoku solving).
6. **Analytics of the Source Code**: Detailed model pipeline visualization for Llama 3 vs. Cloud model. Latency breakdown for different query types (short vs. long responses).
7. **Results Analysis**: Define metrics: Code Generation Accuracy (Exact match %), Logic Task Performance (%), Compute Efficiency (GPU memory, power consumption).
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team

members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies.

Project 25: Evaluating Local Gemma LLM vs. Cloud-Based Model

1. **Title of the Project**: Gemma LLM: How Effective is a Locally Deployed AI Compared to Cloud-Based LLMs?
2. **Literature Overview**: Gemma LLM is Google's lightweight model optimized for on-device execution. It emphasizes efficiency for laptops and small-scale GPU setups. For more details, refer to the paper "Gemma LLM: Google's Efficient Open Model".
3. **Source of the Code**: Gemma GitHub
4. **Choice of the Platform**: Local Deployment: Run Gemma via Hugging Face Transformers or Ollama. Cloud-Based Model: Compare against Google Gemini Pro API.
5. **Dataset Management and Acquisition**: Use the same benchmark dataset but focus on legal document summarization and scientific paper abstract generation.
6. **Analytics of the Source Code**: Create visual representation of model components (Gemma vs. Gemini). Evaluate GPU memory requirements for inference.
7. **Results Analysis**: Metrics for evaluation: Summarization Accuracy (ROUGE score, BERTScore), Computation Cost ($ per 1M tokens), Model Load Time (local vs. API call delay).
8. **Conclusion**: Prepare a public video presentation detailing the project's objectives, methodology, results, and individual contributions. Recording via Microsoft Teams is recommended, and ensure the video is submitted by the deadline. All project team members should speak in the video - there is no time limitation on video. Be aware that the video will go public to increase your position among stakeholders in companies