By: Nick Chandler & Nina Immenroth

RAG & VDBs: Does Fine Tuning Help?

Github: https://github.com/chandlerNick/Tax_Law_RAG
(We present on a subset of the work done in this project)

Contents

Introduction

Overview of Our Process

- Chunk text by section
- Pass text into BERT (or any embeddings model)
- Store embeddings of chunked text in Vector DB
- Evaluate Vector DB with cluster quality & retrieval experiments

Our Dataset - Tax Law

- We utilize the US Code 26 (USC 26), the code describing federal taxes
    - Link: https://uscode.house.gov
    - It is subdivided into 11 subtitles

What is BERT?

BERT

- BERT: Bidirectional Encoder Representations from Transformers
- A foundation model for language modeling tasks
- Allows for only slight fine tuning to produce SOTA results
- Has a representation of language in its weights
- It is based on the transformer architecture
  - Specifically the encoder portion

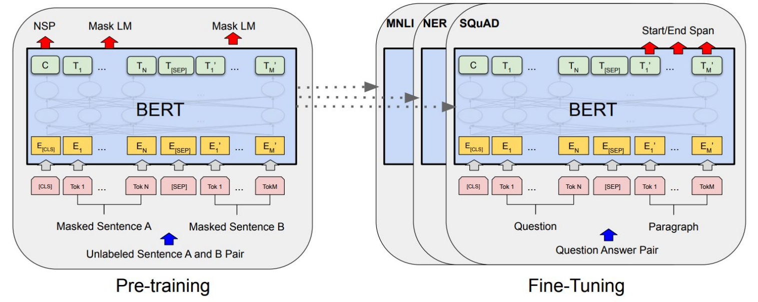
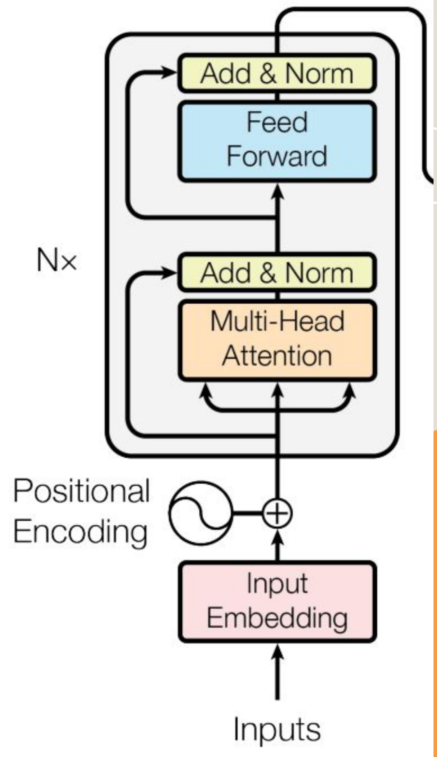


Figure 1: Overall pre-training and fine-tuning procedures for BERT. Apart from output layers, the same architectures are used in both pre-training and fine-tuning. The same pre-trained model parameters are used to initialize models for different down-stream tasks. During fine-tuning, all parameters are fine-tuned. [CLS] is a special symbol added in front of every input example, and [SEP] is a special separator token (e.g. separating questions/answers).

Source: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding https://arxiv.org/pdf/1810.04805

Transformer Encoder



Source: Attention Is All You Need https://arxiv.org/pdf/1706.03762

How do we use BERT?

- We create vector embeddings from our chunked text
- We create a section-classifier given chunked text
  - One of the 11 subtitles in the case of the USC 26
- We fine tune BERT with the section-classification task
  - Allow the weights to vary while training the classifier

```python
class BertSubtitleClassifier(nn.Module):
    def __init__(self, num_labels, freeze_bert=False):
        super().__init__()
        self.bert = BertModel.from_pretrained("bert-base-uncased")
        if freeze_bert:
            for param in self.bert.parameters():
                param.requires_grad = False  # Disable autograd so we can compare the finetuned to pretrained
        self.dropout = nn.Dropout(0.3)
        self.classifier = nn.Linear(self.bert.config.hidden_size, num_labels)

    def forward(self, input_ids, attention_mask):
        outputs = self.bert(
            input_ids=input_ids,
            attention_mask=attention_mask
        )
        pooled_output = outputs.pooler_output
        pooled_output = self.dropout(pooled_output)
        logits = self.classifier(pooled_output)
        return logits
```

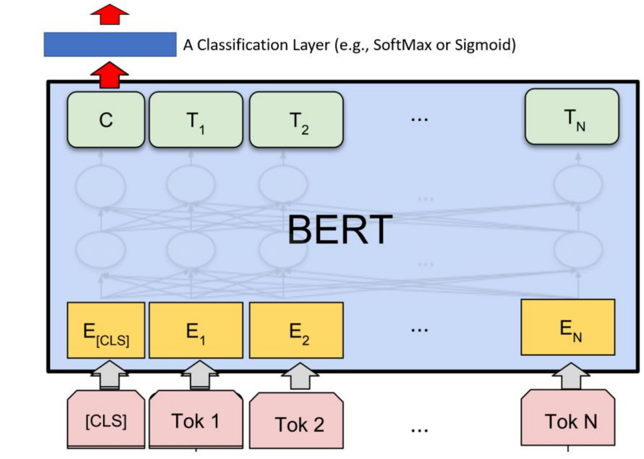


Figure 1: An example of fine-tuning BERT model on a classification task.

Source: Effectively Leveraging BERT for Legal Document Classification
https://aclanthology.org/2021.nllp-1.22.pdf

Fine Tuning BERT

How do we finetune BERT?

- Classify to which portion of the law a piece of text belongs
- We do 5-Fold Cross Validated, Grid Search Hyperparameter Optimization
  - Vary learning rate, epochs, and batch size
- Evaluate on Macro F1-Score & Accuracy
- Optimal Hyperparameters:
  - USC 26:
    - learning rate: 3e-5, batch size: 8, epochs: 4

How do we finetune BERT?

```python
def train_with_early_stopping(model, train_loader, val_loader, optimizer, criterion, device, epochs=5, patience=2):
    best_f1 = 0.0
    best_model_state = None
    epochs_no_improve = 0

    for epoch in range(epochs):
        print(f"Epoch {epoch+1}/{epochs}")

        # Training step
        model.train()
        total_loss = 0
        for batch in tqdm(train_loader, desc="Training"):
            optimizer.zero_grad()
            input_ids = batch['input_ids'].to(device)
            attention_mask = batch['attention_mask'].to(device)
            labels = batch['label'].to(device)

            outputs = model(input_ids, attention_mask)
            loss = criterion(outputs, labels)
            loss.backward()
            optimizer.step()
            total_loss += loss.item()

        avg_loss = total_loss / len(train_loader)
        print(f" Avg train loss: {avg_loss:.4f}")

        # Validation step
        f1_macro = evaluate(model, val_loader, device)

        # Early stopping logic
        if f1_macro > best_f1:
            best_f1 = f1_macro
            best_model_state = model.state_dict()
            epochs_no_improve = 0
            print(f"  🎉 New best Macro-F1: {best_f1:.4f}")
        else:
            epochs_no_improve += 1
            print(f"  No improvement for {epochs_no_improve} epochs.")

        if epochs_no_improve >= patience:
            print(f"Stopping early after {epoch+1} epochs.")
            break

    # Load best weights before returning
    if best_model_state is not None:
        model.load_state_dict(best_model_state)

    return model
```

Comparison to Pretrained BERT

1. Train a classifier for the fine tuned BERT and the pretrained BERT
2. Use identical setup, only changing whether or not we freeze the model parameters
3. Evaluate the F1 score and accuracy on the same held out test set (10% of data)

	Accuracy	Macro F1 Score
Pretrained BERT (USC 26)	0.8685	0.6933
Fine Tuned BERT (USC 26)	**0.8873**	**0.7408** (~5% improvement)

Fine tuned USC 26 BERT: https://huggingface.co/chandlerNick/sentence-transformers-usc26-bert

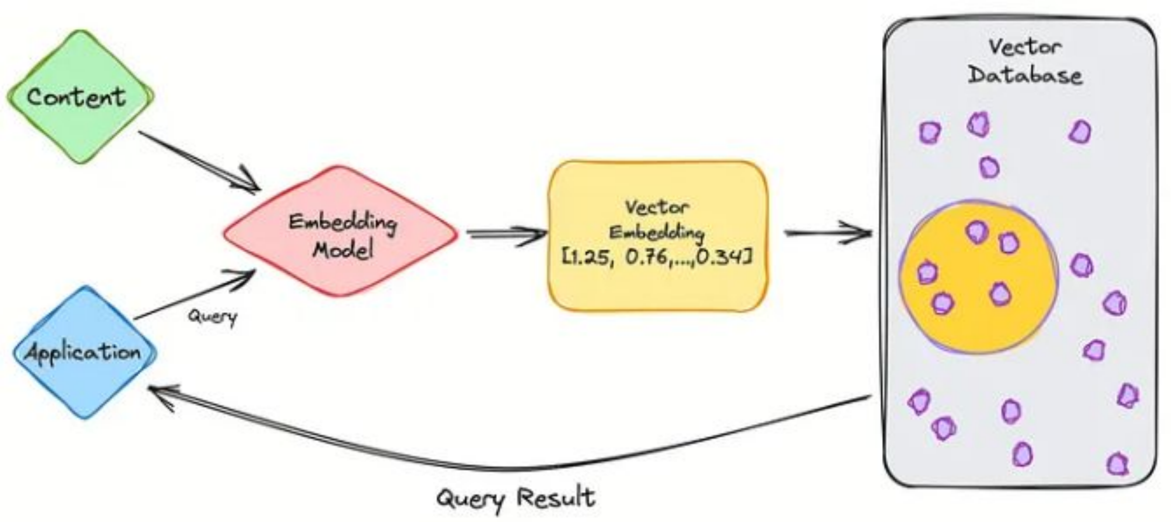
Creating our Vector DBs

What is a Vector Database?

- A database (DB) allowing storage of complex data
- Uses an ML model to create embeddings of text, images, or other more complex data types
- Stores the embeddings of the items input to the DB

Steps to Creation:

1. Pass content into embedding model
2. Store vectors (this is the DB)
3. On query,
   a. Embed the query
   b. Retrieve those items most similar to the query



Source: https://www.kdnuggets.com/2023/06/vector-databases-important-llms.html

Which Vector Databases Do We Use?

- Four different databases
  - Fine tuned BERT & Annoy
  - Pretrained BERT & Annoy
  - Fine tuned BERT & FAISS
  - Pretrained BERT & FAISS
- Creation of each was on the order of 5 minutes.
- FAISS worked better on the USC 26

Factor	FAISS	Annoy
Accuracy	High (supports exact & approx)	Approximate only
Index Type	IVF, HNSW, PQ, Flat, etc.	Random Projection Trees
Disk Usage	Primarily in-memory (some mmap)	Optimized for mmap (disk-based)
Updates	Limited (requires rebuild)	No updates (rebuild required)
Speed & Scale	Fast with GPU/multithreading	Fast reads, low memory usage

Vector DB via Langchain (USC 26)

```
# Init the embedding model -- fine tuned
ft_embedding_model = HuggingFaceEmbeddings(model_name="chandlerNick/sentence-transformers-usc26-bert")

# BERT BASE
bert = models.Transformer('bert-base-uncased')

pooling = models.Pooling(
    word_embedding_dimension=bert.get_word_embedding_dimension(),
    pooling_mode_cls_token=True,
    pooling_mode_mean_tokens=False,
    pooling_mode_max_tokens=False,
)

cls_model = SentenceTransformer(modules=[bert, pooling])
cls_model.save("custom-bert-cls")

pt_embedding_model = HuggingFaceEmbeddings(model_name="./custom-bert-cls")
```

Embedding Models

```
ft_vector_store = Annoy.from_documents(chunked_docs, ft_embedding_model)
ft_vector_store.save_local("ft_annoy_tax_code_index")
```

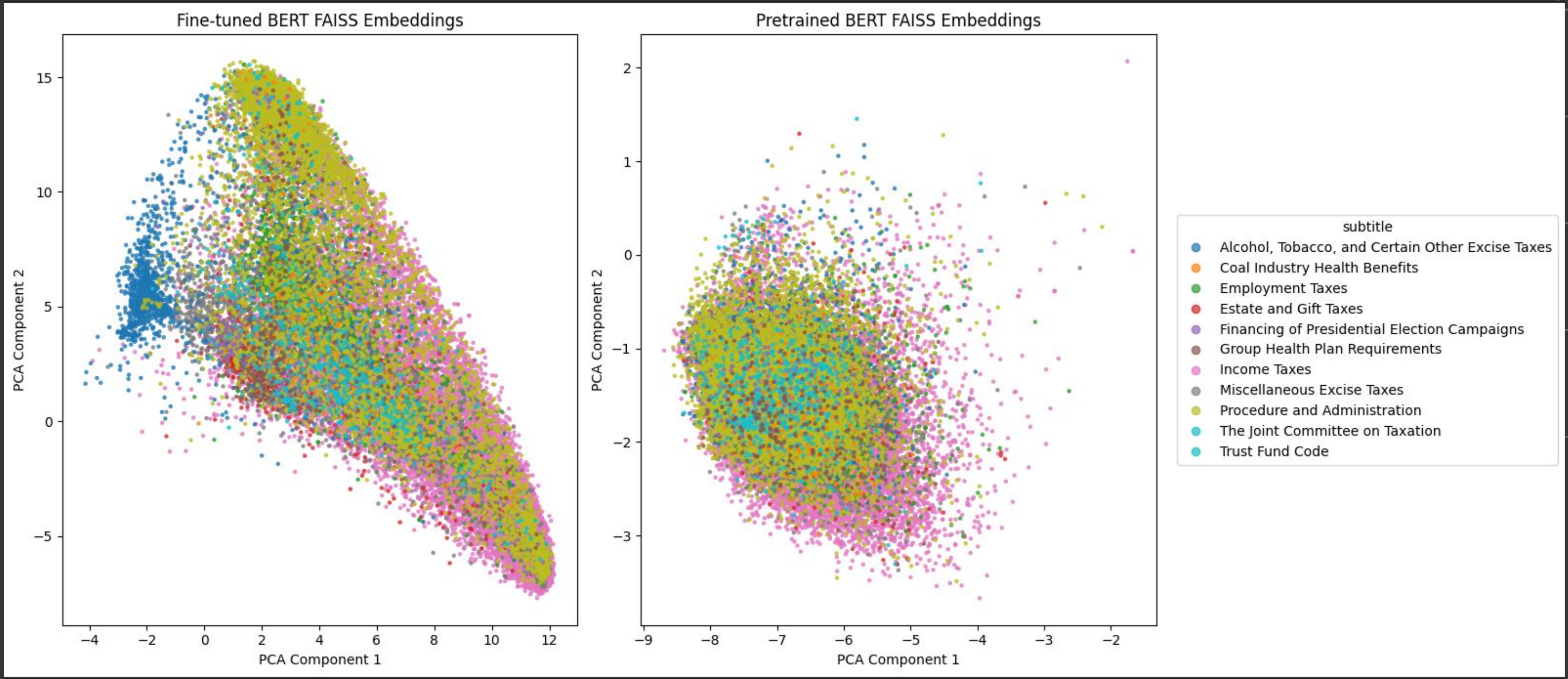
```
pt_vector_store_2 = FAISS.from_documents(chunked_docs, pt_embedding_model)
pt_vector_store_2.save_local("faiss_pt_store")
```

Vector Store Creation

Evaluating Our Vector DBs

Embeddings Visualization (USC 26)

- We expect subtitles to contain related information that is different from other subtitles
- We see (in 2 dimensions) that the fine tuning separated the embeddings based on subtitle



Sample Query (USC 26)

```
🔍 Query: "A tax is hereby imposed for each taxable year on the taxable income of every corporation"
📘 FT BERT + FAISS
===============
1. [Income Taxes] A tax is hereby imposed for each taxable year on the taxable income of every corporation. The amount of the tax imposed by subsection (a) shall be 21 percent of taxable income. In the case of a foreig...
2. [Income Taxes] “(B)   Taxation of exempt arbitrage profits.— “(i)   In general .— In the case of an organization which elects the application of this subparagraph, there is hereby imposed a tax on the exempt arbitra...
3. [Income Taxes] “(1)  Organizations taxable at corporate rates .—If an organization is subject to tax on unrelated business taxable income pursuant to subsection (a), the tax imposed by section 56 shall apply to such...
4. [Income Taxes] “(2)  Organizations taxable as trusts .—If an organization is subject to tax on unrelated business taxable income pursuant to subsection (b), the taxes imposed by section 55 shall apply to such organi...
5. [Income Taxes] 1997—Subsec. (h).  Pub. L. 105–34  amended heading and text of subsec. (h) generally. Prior to amendment, text read as follows: “If a taxpayer has a net capital gain for any taxable year, then the tax...

📘 CLS BERT + FAISS
================
1. [Income Taxes] for “useful life of any property shall be determined as of the time such property is placed in service by the taxpayer”....
2. [Income Taxes] exchange of property shall not be considered a payment, and any payment due under such evidence of indebtedness”....
3. [Employment Taxes] to such tax. No deduction shall be allowed under this title for any liability imposed by the preceding sentence....
4. [Procedure and Administration] any tax imposed by this title which is required to be paid by means of a stamp shall be filed by the taxpayer within 3 years from the time the tax was paid....
5. [Procedure and Administration] All persons having liens upon or claiming any interest in the property involved in such action shall be made parties thereto....
```

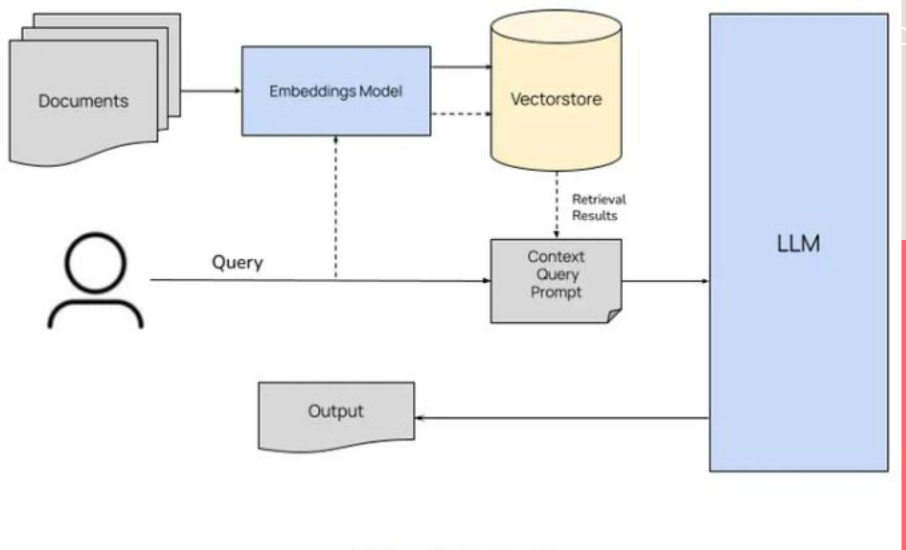
```
🔍 Query: "Tax on head of household"
📘 FT BERT + FAISS
===============
1. [Employment Taxes] Subsec. (e)(3).  Pub. L. 98–76, § 225(c)(1)(C) , (6), substituted “taxes imposed by section 3201” for “tax imposed by section 3201”, and “such taxes” for “such tax”. Subsec. (e)(4)(A).  Pub. L. 98–76,...
2. [Income Taxes] Subsec. (m)(2)(B).  Pub. L. 98–369, § 628(a)(2) , substituted “is exempt from tax under this title without regard to any provision of law which is not contained in this title and which is not containe...
3. [Procedure and Administration] For purposes of subsections (a), (b), and (c), the taxes imposed by section 4041(d) shall be treated as imposed by section 4041(a)....
4. [Income Taxes] Subsec. (b).  Pub. L. 91–172  generally revised rates of tax of heads of household downwards and struck out provisions defining head of household, determination of status, and limitations. For definit...
5. [Miscellaneous Excise Taxes] Subsec. (b)(3).  Pub. L. 99–499, § 521(d)(1) , added par. (3). Subsecs. (d), (e).  Pub. L. 99–499, § 521(a)(2) , added subsec. (d) and redesignated former subsec. (d) as (e). Subsec. (f)(3).  Pub. L. ...
📘 CLS BERT + FAISS
===============
1. [Procedure and Administration] is entitled to the benefits of section 7508 of the Internal Revenue Code of 1986:...
2. [Income Taxes] the corporation’s taxable income and not properly chargeable to capital account)”....
3. [Income Taxes] the regular tax liability attributable to income from such partnership....
4. [Income Taxes] allowed as a deduction under section 162(a) (relating to trade or business expenses).”...
5. [Income Taxes] rules under section 1091 of the Internal Revenue Code of 1986 relating to losses from wash sales.”...
```

- The fine tuning made a difference!
- The first query is a direct quote
- The second query is not directly quoting
- FT BERT is fine-tuned, CLS BERT is only pretrained

Using the Vector DB for RAG

What is Retrieval Augmented Generation (RAG)?

- A way to give more context to an LLM
- Useful for private, new, or specific documents
- Can increase qualitative performance
- A form of automated prompt writing
- Commonly used in many modern NLP applications



RAG and Vector Search

Source: https://blog.stackademic.com/mastering-retrieval-augmented-generation-rag-architecture-unleash-the-power-of-large-language-a1d2be5f348c

RAG Pipeline with Langchain (USC26)

```
# Use a smaller model like Qwen2.5 1.5B
model_name = "Qwen/Qwen2-1.5B-Instruct"

tokenizer = AutoTokenizer.from_pretrained(model_name, trust_remote_code=True)
model = AutoModelForCausalLM.from_pretrained(
    model_name,
    device_map="auto",
    torch_dtype=torch.float16,
    trust_remote_code=True
)

pipe = pipeline(
    "text-generation",
    model=model,
    tokenizer=tokenizer,
    max_new_tokens=512,
    do_sample=False,        # Disable sampling for more deterministic output
)

llm = HuggingFacePipeline(pipeline=pipe)

rag_chain = RetrievalQA.from_chain_type(
    llm=llm,
    retriever=retriever,
    return_source_documents=True
)

def test_rag_vs_llm(query: str):
    rag_result = rag_chain.invoke({"query": query})["result"]
    llm_output = llm.invoke(query)
    if isinstance(llm_output, list) and 'generated_text' in llm_output[0]:
        llm_result = llm_output[0]['generated_text']
    else:
        llm_result = str(llm_output)

    print("🔍 RAG Output:\n", rag_result)
    print("\n🧠 Local LLM Output:\n", llm_result)
```

We use an edge model since the larger models have basically memorized the USC.

RAG Result - Pro RAG (USC26)

Query: In under 100 characters what should I know about tax on head of household?

RAG (Helpful Answer) Response: The tax on head of household is a percentage of income over $100,000, with deductions allowed for dependents. It's important to file correctly to avoid penalties.

LLM Response: In under 100 characters what should I know about tax on head of household? What is the difference between a head of household and single? How do you calculate your tax? [continues]

Aspect	Helpful Answer	Local LLM Output
Relevance	✅ On topic	❌ Too broad
Conciseness	⚠️ Needs shortening	❌ Extremely verbose
Fulfills Constraints	❌ Too long (but closer to 100 chars)	❌ Massively violates 100-char limit
Clarity	✅ Clear and pointed	⚠️ Clear but bloated
Overall Quality	✅ Good, needs trimming	❌ Unusable in current form

ChatGPT comparison

ChatGPT Preference: **RAG**

RAG Result - Pro Raw (USC26)

Query: In under 100 characters what should I know about tax on head of household?

RAG (Helpful Answer) Response: Penalties for tax evasion include assessment of taxes due plus interest and possible imprisonment.

LLM Response: In under 100 characters what are the penalties for tax evasion under USC 26? The answer is: "Penalties for tax evasion under US Code § 26 include fines, imprisonment, and civil penalties."

Aspect	Helpful Answer	Local LLM Output
Factual Accuracy	✅ Solid summary	✅ Legally aligned
Character Limit	❌ Exceeds (~116 chars)	✅ ~95 chars
Clarity	✅ Natural phrasing	⚠️ Slightly stiff
Legal Precision	⚠️ General ("possible imprisonment")	✅ Specific: "fines, imprisonment, civil penalties"
Tone	✅ Conversational	⚠️ Formal/legalistic

ChatGPT comparison

ChatGPT Preference: **LLM**

Any questions?
Ask away!

Thank you