NeviTech şirketi için Chatbot

Giriş

Bu rapor, şirket için LLM kullanılarak geliştirilen bir chatbot projesinin geliştirilme aşamalarıyla ilgilidir. Geliştirme aşamasında kullanılan yöntemler, teknolojiler ve modeller açıklanmıştır.

Kullanılan Yapay Zeka ve 'Gömme' modelleri

- Mistral AI/Mistral-7B-Instruct-v0.3: Bu model, Mistral AI şirketinin 22 Mayıs tarihinde kullanıma açtığı 7.25 milyar parametreli modelidir. Hızlı, küçük ve yüksek performansından ve Apache 2.0 lisansından dolayı seçilmiştir.

- Nomic-ai/nomic-embed-text-v1.5: Bu gömme modelini seçme sebebim çok popüler olması ve küçük olması.

- GritLM/GritLM-7B: Bu gömme modeli, HuggingFace MTEB sıralamasında an itibari ile 7. sırada.

- intfloat/e5-mistral-7b-instruct: Bu gömme modeli, HuggingFace MTEB sıralamasında an itibari ile 8. sırada.
    - `e5-mistral-7b-instruct` modeli genelde İngilizce için kullanılıyor, eğer birçok dil için kullanacaksak, bu modelin biraz daha büyük versiyonu olan `intfloat/multilingual-e5-large` kullanılabilir.

Kullanılan Python kütüphanesi

- **Llama-Index:** Llama-Index, yeni ve topluluğu her geçen gün daha da büyüyen bir veri framework'üdür. Kullanma sebebim kullanmasının daha kolay ve düzenli olmasıdır.

Eklediğim Fonksiyonlar

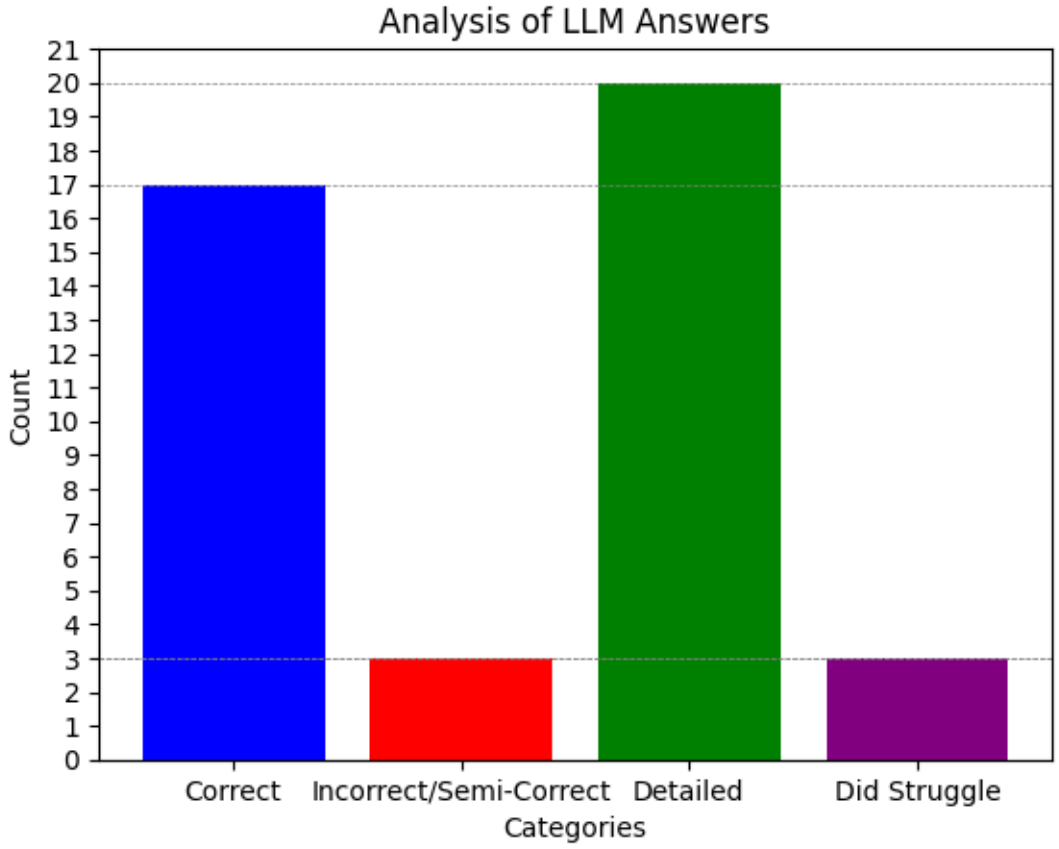
- **Excel Reader:** Llama-Index kütüphanesinde çoğu dosya formatını işleyebiliyoruz, ama maalesef Excel formatı bunlardan biri değil, excel dosyalarını Yapay Zeka modelinin anlayacağı şekilde işlemek ve modele sunmak için kendi tool'umu yazdım. Bu şekilde, excel dosyalarını hem rahatça okuyabilecek, hem de yapay zekaya uygun formatta sunarak, yapay zekanın bu bilgileri rahatlıkla işleyebilmesine olanak sağlayacak.

```python
class ExcelReader(BaseReader):
    def load_data(self, file_path: Path, extra_info: Optional[Dict] = None) -> List[Document]:
        if extra_info is not None:
            if not isinstance(extra_info, dict):
                raise TypeError("extra_info must be a dictionary.")

        df = pd.read_excel(file_path)
        # Create a list to store the documents
        documents = []

        # Iterate over each row in the DataFrame
        for _, row in df.iterrows():
            # Convert the row to a dictionary
            row_dict = row.to_dict()
            # Serialize the dictionary to a JSON-formatted string
            json_string = json.dumps(row_dict, indent=4)
            # Create a Document for each JSON string
            documents.append(Document(text=json_string, metadata=extra_info))

        return documents
```

- **Hybrid Retriever:** Llama Index gibi frameworkler yapay zeka modellerinin dosyalardan bilgi alması için 'getirici/retriever' diye adlandırılan sistemi kullanırlar. En popüler ve kullanışlı olanlarından bazıları 'anahtar kelime arama' ve 'vektör arama' denilen tekniklerdir, ben ise bu iki tekniği birbirleriyle entegre ederek dosyalardan daha yüksek doğruluk oranı ile bağlantılı bilgiler alabildim.

Test Aşaması

Programı test etmek için 20 tane soru oluşturdum, soruları ve yapay zekanın cevaplarını test.txt dosyasına kaydettim, ardından bir grafik oluşturarak soruların doğru, yanlış ve ayrıntılı olup olmama gibi kategoriler üzerinde analizini yaptım. Sorulara ve analize buradan erişebilirsiniz.



Bu tabloya bakılarak soruların %85'ine doğru cevap verdiğini ve tüm cevapların detaylı olduğunu görebiliriz.

20 sorudan 17'sinde doğru, 2'sinde yanlış ve 1'inde de yarı doğru cevap verdi, 17-18-19. Sorularda cevap süresi diğer cevaplar için gereken süreden daha uzundu.

M.Cihan Yalçın - 11.06.2024