Web Trafik Loglarına Dayalı Yapay Zeka Destekli Soru-Cevap Sistemi Geliştirme

Furkan Küçük

18/08/2024

Contents

Yönetici özeti

Geliştirdiğim program, Apache web server tarafından üretilen loglardan önemli bilgileri filtreleyerek, filtrelenen bilgileri Pinecone vektör veritabanına yükler. Ardından gelen kullanıcı sorgusundan istenilen bilgileri NLP, regex ve filtreleme yardımıyla çıkartır, çıkartılan bilgileri kullanarak vektör veritabanından bilgi çeker, çekilen bilgilerin doğruluğunu arttırmak amacıyla yeniden filtreleme işlemi yapılır ve en son aşamada elde edilen bu bilgileri de GPT2 modelini kullanarak doğal dile çevirir ve kullanıcıya iletir.

Nasıl kullanılır

Programı kullanmak için, program çalıştırıldıktan sonra gelen "Ask me anything:" sorusuna aşağıdaki gibi sorgular göndermeniz gerekmektedir.

"Retrieve all user data from August"

"Retrieve all user data for IP: 127.0.0.1"

“Retrieve all user data from 17 August 2024, visited page: phpmyadmin/"

"Gather all data from August 17"

"Get all user information for visited page: phpmyadmin/"

Program, aşırı gelişmiş bir entity extraction metotu kullanmadığından ötürü, spaCy'nin NLP işlemi bazı sorgulardan önemli bilgileri çıkartmakta yetersiz kalabilir, istediğiniz cevapları alamadığınız beklenmedik sonuçlar ortaya çıkabilir.

Apache logları IPv6 adreslerini tuttuğu için, localhost “::1” olarak gözükmektedir.

Detaylı özet

Log ayrıştırma ve veritabanına yükleme:

Apache tarafından üretilen logları, örneğin:

::1 - - [12/Aug/2024:15:22:33 +0300] "GET /dashboard/ HTTP/1.1" 200 5187 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36 OPR/111.0.0.0"

inceleyerek, içerisinden IP adresi (::1), gün/ay/yıl (12/Aug/2024), ziyaret edilen sayfa (/dashboard/) gibi önemli bilgileri ayrıştırır. Ayrıştırılan bu bilgiler, "Day of month: {day}, Month of year: {month}, Year: {year}, IP address: {ip}, Visited page: {page}" formatıyla birleşik bir String haline getirilir, ardından bu String Pinecone veritabanına vektörleştirilerek yüklenir.

Veritabanından bilgi çekme:

Kullanıcı tarafından gönderilen sorgu, spaCy kütüphanesinin "en_code_web_sm" adlı modeliyle, NLP mantığıyla işlenir. İşlenen bu sorgudan, NLP yardımı ve NLP'nin yetersiz kaldığı yerlerde bazı basit regex ve filtreleme yardımıyla önemli bilgiler çıkartılır ve bir Dictionary içerisine kaydedilir.

Örneğin, kullanıcı tarafından gönderilen sorgu "Retrieve all user data from 17 August 2024, visited page: "/dashboard/"" ise,

{"IP_address": None, "Day_of_month": 17, "Month_of_year": Aug, "Year": 2024, "Visited_page": "/dashboard/"} şeklinde Dictionary içerisine yazılır.

Ardından sorgudan çıkartılan bu bilgileri, Pinecone veritabanına yükleme yapılırken kullanılan formata sokarak,

("Day of month: {day}, Month of year: {month}, Year: {year}, IP address: {ip}, Visited page: {page}") yeni bir String oluşturulur, ve oluşan bu String ile Pinecone veritabanına query atılır ve en alakalı 20 veriyi çekmesi istenir.

Çekilen bilgiyi doğrulama:

Önceki adımlarda veritabanından çekilen veri, programın doğruluğunu arttırmak amacıyla basit bir filtreleme işlemine sokulur.

Çıkartılan Dictionary içerisindeki bilgiler, Pinecone veritabanından çekilen bilgilerde bulunuyor mu diye kontrol edilir, eğer bulunuyorsa "filtered_logs" listesine kaydedilir, bulunmuyorsa program bu veriyi listeye eklemez.

Doğrulanan bilgileri kullanıcıya gönderme:

Doğruluğu onaylanan bilgileri tekrardan formatlayarak tek bir değişken içerisine konulur, ardından GPT2 modeline bu değişken gönderilir ve doğal dile çevrilmesi istenir, GPT2'nin yanıtını alarak doğal dile çevrilen bu mesaj, kullanıcıya gönderilir.

Performans Değerleri:

Program, Precision, Recall ve F1 Score olarak 3 farklı performans hesaplaması yapar.

Precision, filtreleme sonrasında doğruluğu onaylanan verilerin, Pinecone'dan çekilen bütün verilere oranıdır.

(matched_logs / total_logs)

Recall, filtreleme sonrasında doğruluğu onaylanan verilerin, Pinecone veritabanı içerisinde bulunan bütün alakalı verilere oranıdır.

(matched_logs / relevant_logs)

(Program default olarak Pinecone'dan 20 veri çeker, ancak Pinecone veritabanı içerisinde, sorguyla alakalı olan 20'den fazla veri bulunabilir, dolayısıyla Precision ve Recall değerleri farklı çıkabilir.)

F1 Score, Precision ve Recall sonuçlarının harmonik ortalamasıdır, hem Precision hem de Recall sonuçlarını dengeleyen tek bir metrik sağlar.

(2 * (precision * recall) / (precision + recall))

Karşılaşılan zorluklar

Log Dosyalarının Ayrıştırılması

Apache log dosyalarının formatı, her zaman tutarlı olmayabiliyor. Bazı log satırları eksik bilgiler içerdiği için bu satırları doğru bir şekilde ayrıştırmakta zorluklar yaşadım. Bu sorunu aşmak için, satırları düzgün bir şekilde ayrıştıramadığım durumlarda, satırları atlamayı ve yalnızca tam verileri kullanmayı tercih ettim.

Entity Extraction

Önemli bilgileri çıkartmak amacıyla ilk başta T5 modelini kullanmayı denedim, ancak "t5-small" veya "t5-base" modellerinin hiçbiri istediğim sonuçları bana vermedi, kullanıcının gönderdiği sorgudan benim ihtiyacım olan IP adresi, tarih gibi bilgileri doğru çıkartmasını sağlayamadım.

Ardından spaCy ve NLP kullanmaya başladım, spaCy modelini kullanarak metinlerden gerekli bilgileri çıkartmak her zaman mükemmel sonuçlar vermedi, ancak T5'e oranla daha iyi sonuçlar alıyordum.

Özellikle tarih, IP adresi ve ziyaret edilen sayfa gibi spesifik bilgilerle ilgili sorunlar yaşandı. Örneğin spaCy'nin sorudaki IP adresini "e:" gibi cümleden rastgele seçilmiş birkaç karakter olarak belirlemesi, programın geri kalan kısmının düzgün çalışmasında büyük sorunlar çıkarttı. Bu durumla başa çıkmak için regex ve manuel kontrol mekanizmaları geliştirdim fakat bu yöntemler de bazen yetersiz kalıyor ve doğru algılanmayan sorular ortaya çıkabiliyor.

Performans Hesaplamaları

Precision, Recall ve F1 Score gibi performans metriklerini hesaplamak için doğru ve alakalı veri setlerini oluşturmak zorlu bir süreçti.

Pinecone'dan çekilen verilerin, tam olarak kullanıcının aradığı bilgilerle ne derece uyuştuğunu ölçmek için ekstra filtreleme ve doğrulama adımları eklemem gerekti.

Doğal dile çevirme

Kullanıcıya yanıt olarak gönderilecek doğal dildeki metni hazırlamak için farklı yollar denedim, sürekli hatalarla karşılaştım. T5 modelini kullandığımda istediğim metnin oluşturulması görevini yerine getiremedi, aynı sorunları GPT'de de yaşadım, yaptığım araştırmalar sonucu nasıl kullanmam gerektiğini daha iyi kavradım ve programın son halinde çalışır hale getirmeyi başardım.

Başarısız metotlar

Entity Extraction:

Kullanıcı tarafından gönderilen sorgulardan, hatasız bir şekilde veri alabilmek için Groq API kullanarak, sorulan soruyu bir yapay zekaya göndererek verileri çıkartmasını istedim. Ardından bu yaklaşımın proje isterlerince kabul görmeyeceğini öğrendim ve başka yollar ararken T5'e geçiş yaptım.

T5 denemesi:

T5 ile doğal dildeki bu sorguyu incelemesi ve ihtiyacım olan bilgileri çıkartması için "t5_small" ve "t5_base" LLM'lerini kullandım, ancak neredeyse sorguların %90'ında istediğim verileri çıkarmakta başarısız oldum.

T5 fine-tuning:

T5 ile yaşadığım sorunları çözebilmek adına kendim "source" ve "target" şeklinde "sorulan sorgu" ve "çıkartılması gereken bilgi" şeklinde 500 satır sorgu ve çıkartılmış bilgi içeren bir dosya hazırladım, bu dosyayı kullanarak T5 modelini fine tune etmeye çalıştım, bu süreç içerisinde vaktimin limitli ve elimdeki datasetin bilgisayarımın performansına oranla büyük olması sebebiyle TrainingArguments (learning rate, batch size, epochs) kısmını, daha hızlı sonuç alabileceğim bir şekilde ayarladım. Ancak tahminimce fine-tuning'den istediğim sonucu alamamamın en büyük sebeplerinden birisi benim bu süreci çok hızlandırmaya çalışmam oldu, eğer fine-tuning işlemine gerektiği kadar vakit ayırabilseydim daha iyi sonuçlar alabileceğimi düşünüyorum.

Toplamda 6 saat süren bu fine tuning işleminden sonra bile T5 modelinden istediğim yanıtları alamadığımda spaCy NLP mantığına geçiş yaptım.

Doğal dilde yanıt oluşturma:

Doğal dilde yanıt oluşturabilmek için de ilk başta Groq API ile yapay zekaya elde ettiğim verileri gönderiyordum, ardından çıktıyı kullanıcıya gönderiyordum.

Aynı sebepten ötürü bu yaklaşımdan da vazgeçmek zorunda kaldım, GPT2 modeline geçiş yaptım. GPT2 modelinde de istediğim sonuçları alamayınca, elimdeki bütün verileri mantıklı cümleler ve tek bir String haline getirerek GPT2'ye göndermeyi denedim, bu işlem istediğim gibi sonuç verdi.

Performans

```
Time taken to generate the answer: 3.65 seconds
Performance Evaluation:
- Total logs retrieved: 20
- Matched logs: 10
- Relevant logs: 45
- Precision: 0.50
- Recall: 0.22
- F1 Score: 0.31
```

```
Time taken to generate the answer: 0.84 seconds
Performance Evaluation:
- Total logs retrieved: 20
- Matched logs: 5
- Relevant logs: 15
- Precision: 0.25
- Recall: 0.33
- F1 Score: 0.29
Ask me anything:
```

Program, performansı ölçmek için çok basit bir mantık kullanıyor. Kullanıcı soruyu sorduğu andan itibaren, sorunun cevaplanmasına kadar geçen süreyi ölçüyor ve bu süreyi performansı ölçmek amacıyla kullanıyor.

Gelecek Planları

Programı daha da geliştirmek, doğruluğunu arttırmak, kullanıcı sorgularını daha iyi algılamak adına gelecekte yapılabilecek bazı çalışmalar:

Fine-tuning:

Şu anda kullanmakta olduğum spaCy NLP metotundan istediğim verimi alamamam sebebiyle, daha önceden denediğim ancak zaman limiti nedeniyle hızlandırmaya çalıştığım T5 fine-tuning işlemini, hızlandırmak zorunda kalmadan, daha büyük bir dataset ile, saatler hatta günler sürecek bir tuning işlemine sokarak kullanıcı sorularını çok daha iyi algılamasını sağlayabilir, şu anda fail-safe olarak kullanmakta olduğum regex ve manuel soru algılama metotlarını devredışı bırakarak tamamen T5 tabanlı bir soru algılama metotuna geçilebilir. Yeterli zaman ve yeterince kapsamlı bir dataset sağlandığı durumda fine-tuning işleminin çok daha iyi sonuç vereceğine inanıyorum.

Program doğruluğunu arttırmak:

Program, default olarak Pinecone veritabanından 20 veri çekiyor, bu işlemi bu şekilde yapmak yerine, kullanıcı sorgusunu inceleyerek, veritabanı içerisindeki, sorguyla alakalı bütün verileri çekmesini sağlamak, programdan beklenen doğruluk oranını daha da arttırabilir. 384 boyuta sahip olan vektörler vektör normalizasyonu yapılarak daha düşük boyutlara indirilerek anlamlı veriye ulaşma hızı ve oranı artırılabilir, bu şekilde de "**Curse of Dimensionality**" nin önüne bir nebze geçilmiş olur.

Performans:

Programın içerisinde kullanılan NLP, GPT gibi metotların daha hızlı çalışmasını ve istenilen bilgiyi oluşturmasını/çıkartmasını daha kolay ve hızlıca yapmasını sağlamak için bazı geliştirmeler yapılabilir, örneğin entity extraction için fine-tuned bir LLM modeli kullanmak, eminim ki genel amaçlar için üretilmiş bir LLM modelinden çok daha hızlı istediğimiz sonuçları verecektir, aynı durum kullanıcıya yanıt verirken kullanılan GPT modeli için de geçerli, programın işleyişi için özel olarak tasarlanmış veya özel olarak tune edilmiş bir model sayesinde verilen yanıtların hızını arttırmak mümkün.

Kütüphaneler

time: Zaman gecikmeleri ve zamanlayıcı işlemleri için.
re: Regex kullanarak metin işleme ve entity extraction için.
datetime: Tarih formatlarını işlemek ve dönüştürmek için.
spaCy: Doğal dil işleme ve entity extraction için.
Pinecone: Vektör veritabanı yönetimi için.
Sentence_transformers: Metin vektörleştirme için.
Transformers: GPT-2 modelini kullanmak için.
(Başarısız metotlar) Datasets
(Başarısız metotlar) csv
(Başarısız metotlar) T5

Kaynakça

Apache: https://httpd.apache.org/docs/2.4/
Spacy: https://spacy.io/api/doc
Pinecone: https://docs.pinecone.io/reference/api/introduction
Yaşanılan entegre sorularını çözebilmek için birçok StackOverFlow konusu (Hatalar genellikle Python ve/veya Package versiyon uyumsuzlukları kaynaklıydı)
Aynı zamanda, daha verimli çalışan Regexler oluşturmak için ChatGPT kullandım.