İSTANBUL ÜNİVERSİTESİ CERRAHPAŞA
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

BİTİRME PROJESİ

DAVA SONUÇ TAHMINI VE YAPAY ZEKA DESTEKLI YARGI KARAR DESTEK SISTEMI

Hazırlayanlar : Musa Berke ŞENGÖZ 1306200126
Yağmur GÖKÇEKLİ 1306200012

Danışman : Dr. Öğr. Üyesi Ramiz Görkem BİRDAL

HAZİRAN 2025

ÖNSÖZ

Yapay zeka, zamanımızın en hızlı büyüyen alanlarından biridir. Özellikle doğal dil işleme teknolojisindeki gelişmeler, hukuk gibi zorlu, metin ağırlıklı alanlarda bir devrim anlamına gelmiştir. Hukuki süreçlerin yavaşlığı ve doğru bilgiyi bulmanın zorluğu, yapay zeka destekli sistemlerin ne kadar değerli olduğunu göstermektedir. Bu teknolojilerin hukuki araştırmalarda doğruluğu artıracağı ve adalete erişimi demokratikleştireceği, hayatın her alanında kendini gösterecektir.

Yapay zekanın hukukta bugün karşılaştığı en önemli sorun, uzmanlık gerektiren alanlar için genel amaçlı modellerin yetersizliğidir. Büyük dil modelleri, belirli yasal terminolojiyi, yasal hükümler ve kararlar arasındaki anlamsal ilişkileri ve bir avukatın akıl yürütme kapasitesini tam olarak yansıtmaz. Bu nedenle, bu alandaki en son yaklaşımlardan biri, bu modelleri ince ayar ve benzeri teknikler yoluyla belirli bir alana uyarlayarak uzmanlaştırmaktır. Bu çalışma, bu uzmanlaşma sürecinin bir sistemin doğruluk oranını ve yanıt verme kalitesini ne kadar artırdığını göstermeye çalışacaktır.

Bu çalışma boyunca desteklerini esirgemeyen değerli hocam Sayın Asst. Prof. Ramiz Görkem BİRDAL'a en içten dileklerimle teşekkür ediyorum.

Ayrıca tüm eğitim hayatım boyunca bana yol gösteren, her zaman yanımda olan aileme en içten dileklerimle teşekkürlerimi sunuyorum.

Musa Berke ŞENGÖZ 1306200126
Yağmur GÖKÇEKLİ 1306200012

İÇİNDEKİLER

ŞEKİL LİSTESİ

TABLO LİSTESİ

SEMBOL LİSTESİ

A : Anchor (Çapa) vektörü
P : Pozitif örnek vektörü
N : Negatif örnek vektörü
d(A,P) : Anchor ve Pozitif vektörleri arasındaki mesafe
d(A,N) : Anchor ve Negatif vektörleri arasındaki mesafe
α (alpha) : Marjin (margin) değeri

KISALTMA LİSTESİ

LLM : Büyük Dil Modeli (Large Language Model)
NLP : Natural Language Processing
RAG : Retrieval-Augmented Generation
TF-IDF : Term Frequency-Inverse Document Frequency
TF : Term Frequency
IDF : Inverse Document Frequency
SVM : Destek Vektör Makinesi (Support Vector Machine)
FAISS : Facebook AI Similarity Search
BM25 : Okapi BM25
TCK : Türk Ceza Kanunu
KVKK : Kişisel Verileri Koruma Kanunu
EHK : Elektronik Haberleşme Kanunu
ETK : Elektronik Ticaret Kanunu
API : Uygulama Programlama Arayüzü (Application Program Interface)
JSON : JavaScript Object Notation
GPU : Grafik İşlem Birimi (Graphics Processing Unit)
MRR : Mean Reciprocal Rank

ÖZET

YAPAY ZEKA DESTEKLİ HUKUKİ ARAŞTIRMA VE KARAR DESTEK SİSTEMİ

Hukuk alanı, yapay zeka teknolojilerinin dönüştürücü etkisine giderek daha fazla tanıklık etmekte, özellikle hukuki araştırma süreçlerinde verimliliği ve doğruluğu artırma potansiyeliyle öne çıkmaktadır. Kanunlar, yönetmelikler, içtihatlar ve kurul kararlarından oluşan devasa ve karmaşık metin yığınları içinde doğru bilgiye hızla ulaşmak, hukuk profesyonelleri için en temel zorluklardan biridir. Bu çalışma, bu zorluğun üstesinden gelmek üzere geliştirilmiş yapay zeka destekli bir hukuki asistan sistemini sunmaktadır.

Projenin temelini, Büyük Dil Modelleri (LLM'ler) ve bu modellerin doğruluğunu artırmak için kullanılan Retrieval-Augmented Generation (RAG) mimarisi oluşturmaktadır. RAG mimarisi, LLM'lerin cevap üretirken kendi içsel bilgilerine güvenmek yerine, harici ve güncel bir bilgi kaynağından aldığı belgelere dayanmasını sağlayarak, hukuki uygulamalar için kritik olan güvenilirlik ve kaynak gösterme sorunlarına çözüm getirir.

Standart LLM'ler ve temel RAG sistemleri, genel konularda başarılı olsalar da, hukukun kendine özgü terminolojisini ve metinler arası incelikli anlamsal ilişkileri yakalamakta yetersiz kalabilmektedir. Temel vektör arama yöntemleri, anahtar kelime eşleşmelerini kaçırabilirken, sadece anahtar kelime araması da metnin anlamsal bağlamını göz ardı eder. Bu eksiklikleri gidermek amacıyla, Hibrit Arama ve Yeniden Sıralama (Re-ranking) gibi daha gelişmiş retrieval teknikleri geliştirilmiştir.

Bu çalışmada, Türk Bilişim Hukuku alanında uzmanlaşmış, gelişmiş bir RAG sistemi tasarlanmış ve gerçeklenmiştir. Sistemin bilgi çekme (retrieval) katmanının isabet oranını artırmak amacıyla, anlamsal arama için FAISS vektör indeksi ve anahtar kelime tabanlı arama için BM25 indeksi kullanılarak Hibrit Arama altyapısı kurulmuştur. Bu altyapıdan gelen aday sonuçların alaka düzeyini en üst seviyeye çıkarmak için ise Cross-Encoder tabanlı bir Yeniden Sıralama modülü sisteme entegre edilmiştir.

Çalışmanın ikinci kısmında ise, sistemin retrieval performansını daha da iyileştirmek amacıyla, genel amaçlı bir embedding modelinin (intfloat/multilingual-e5-large) hukuki alana özgü verilerle ince ayar (fine-tuning) yapılması süreci incelenmiştir. Bu amaçla, yarı otomatik yöntemlerle özel bir eğitim veri seti oluşturulmuş ve model eğitilmiştir. Eğitilmiş modelin performansı, Recall@K ve MRR gibi standart bilgi erişim metrikleri kullanılarak orijinal modelle karşılaştırılmış ve fine-tuning işleminin, modelin doğru belgeyi üst sıralarda bulma isabet oranını ölçülebilir şekilde artırdığı deneysel olarak kanıtlanmıştır.

SUMMARY

AI-ENHANCED LEGAL RESEARCH AND DECISION SUPPORT SYSTEM

The field of law is increasingly witnessing the transformative effect of artificial intelligence technologies, and stands out with its potential to increase efficiency and accuracy, especially in legal research processes. Quickly accessing the right information in huge and complex textual piles consisting of laws, regulations, precedents and board decisions is one of the most fundamental challenges for legal professionals. This study presents an artificial intelligence-supported legal assistant system developed to overcome this challenge.

The basis of the project is Large Language Models (LLMs) and the Retrieval-Augmented Generation (RAG) architecture used to increase the accuracy of these models. The RAG architecture provides solutions to reliability and citation problems that are critical for legal practices by allowing LLMs to rely on documents obtained from an external and up-to-date information source instead of relying on their own internal knowledge when producing answers.

Standard LLMs and basic RAG systems, although successful in general subjects, may be inadequate in capturing the specific terminology of law and the subtle semantic relationships between texts. While basic vector search methods may miss keyword matches, keyword-only search also ignores the semantic context of the text. In order to overcome these deficiencies, more advanced retrieval techniques such as Hybrid Search and Re-ranking have been developed. In this study, we have designed and implemented an advanced RAG system specialized in the field of Turkish Information Law. In order to increase the hit rate of the retrieval layer of the system, we have established a Hybrid Search infrastructure using the FAISS vector index for semantic search and the BM25 index for keyword-based search. In order to maximize the relevance of the candidate results coming from this infrastructure, we have integrated a Cross-Encoder-based Re-ranking module into the system.

In the second part of the study, we investigated the process of fine-tuning a general-purpose embedding model (intfloat/multilingual-e5-large) with legal domain-specific data in order to further improve the retrieval performance of the system. For this purpose, we created a special training dataset with semi-automatic methods and trained the model. We compared the performance of the trained model with the original model using standard information retrieval metrics such as Recall@K and MRR, and experimentally proved that the fine-tuning process measurably increased the accuracy of the model in finding the correct document at the top.

1. GİRİŞ

Yapay zeka ve doğal dil işleme teknolojileri, günümüzde hukuk gibi metin ağırlıklı ve karmaşık alanlarda devrim niteliğinde dönüşümler yaratma potansiyeli taşıyor. Hukuk profesyonellerinin ve akademisyenlerin, sürekli güncellenen kanunlar, yönetmelikler, tebliğler ve her gün binlercesi eklenen mahkeme kararları gibi devasa bir bilgi yığını içerisinde doğru ve ilgili bilgiye hızla ulaşması, adalet süreçlerinin etkinliği için hayati önem taşımaktadır. Bu çalışma, bu zorluğun üstesinden gelmek amacıyla, özellikle Türk Bilişim Hukuku alanında, yapay zeka destekli bir hukuki araştırma ve soru-cevap asistanı geliştirmeyi hedeflemektedir.

Hukuk alanına sayısal yöntemlerin uygulanmasının zengin bir geçmişi vardır. Alandaki ilk çalışmalar genellikle sonuç tahminine odaklanmıştır. Örneğin, Katz, Bommarito ve Blackman'in çalışması [1], istatistiksel modellere dayanarak ABD Yüksek Mahkemesi'nin kararlarını yüksek bir doğrulukla tahmin etme yeteneğini ortaya koymuştur. Eş zamanlı olarak, Thomson Reuters'ın Westlaw ve RELX'in LexisNexis [2] gibi ticari platformları, basit anahtar kelime aramasından başlayıp daha gelişmiş Doğal Dil İşleme (DDİ) tekniklerini dahil ederek hukuki araştırmalar için vazgeçilmez araçlar haline gelmiştir. Fakat bu güçlü sistemler, temel olarak bir kullanıcının bulduğu belgeler listesinden bilgiyi manuel olarak sentezlemesini gerektiren arama motorları olarak işlev görür. Güncel paradigma değişimi üretken yapay zeka ile yaşanmaktadır. Küresel hukuk büroları tarafından benimsenen Harvey AI ve Casetext'in CoCounsel [3] gibi sistemler, sohbet tabanlı soru-cevap, belge taslağı hazırlama ve metin özetleme gibi yetenekler sunmak için Büyük Dil Modelleri'nden (LLM) yararlanmaktadır. Bununla birlikte, bu sistemler genellikle genel amaçlı LLM'ler üzerine kuruludur ve yoğunlukla Anglo-Amerikan hukuk sistemi için optimize edilmiştir; bu durum, Türk Hukuku gibi spesifik hukuk alanlarına ve dillere özel, uzmanlaşmış araçlara yönelik belirgin bir ihtiyaç doğurmaktadır.

Bu çalışmada, literatürdeki mevcut sistemlerin sınırlılıklarını aşmayı hedefleyen, LexAI adında gelişmiş bir RAG sistemi tasarlanmış ve gerçeklenmiştir. Türk Bilişim Hukuku'nu kapsayan geniş bir bilgi tabanı (kanunlar, yönetmelikler vb.) üzerinde, hem anlamsal arama için FAISS vektör indeksi hem de anahtar kelime tabanlı arama için BM25 indeksi kullanılarak Hibrit Arama altyapısı kurulmuştur. Arama sonuçlarının isabet oranını en üst düzeye çıkarmak için Cross-Encoder tabanlı bir Yeniden Sıralama (Re-ranking) katmanı eklenmiştir. Projenin en özgün yönü ise, genel amaçlı bir embedding modelinin, Türk Bilişim Hukukuna özel olarak "Soru Üret & Zor Negatif Bul" stratejisiyle oluşturulan veri seti ile ince ayar (fine-tuning) yapılarak uzmanlaştırılmasıdır. Sistemin cevap üretme katmanında ise, kendisine sunulan kaynaklara dayalı, gerekçeli, formatlı cevaplar vermesi için detaylı bir promptla yönlendirilen Llama 3.1 8B-Instruct modeli kullanılmıştır.

Bu çalışma, mevcut literatürdeki yerini özgün bir şekilde konumlandırmaktadır. Harvey AI [3] gibi modern uygulamalara benzer şekilde, geniş bir hukuki külliyata dayanarak sohbet tabanlı cevaplar sunmak için bir RAG mimarisi kullanmaktadır. Tahmin sistemlerinden [1] farklı olarak, sonuçları öngörmek yerine gerekçeli, kaynağa dayalı analiz sunmaya odaklanmaktadır. Geleneksel bilgi erişim platformlarından [2] ise, sadece ilgili belgeleri bulmakla kalmayıp, bu bilgiyi tutarlı ve insan tarafından okunabilir bir cevap halinde sentezleyerek bir adım öteye geçmektedir. Bu çalışmanın, genel amaçlı hukuki LLM'lerden [3] ayrılan en önemli ve özgün katkısı, temel embedding modelinin alana özgü olarak uzmanlaştırılmasıdır. intfloat/multilingual-e5-large modeli, Türk Bilişim Hukuku üçlülerinden oluşan özel bir veri seti ile ince ayar (fine-tuning) yapılarak, kendi özel alanında ölçülebilir şekilde daha yüksek bir arama doğruluğuna ulaşmıştır. Bu tez, uzmanlaştırılmış ve fine-tune edilmiş bir RAG sisteminin, niş hukuk alanlarında daha büyük ve genel sistemlerden daha üstün bir performans sunabildiğini Recall@K ve MRR gibi standart metriklerle deneysel olarak kanıtlamaktadır.

2. GENEL KISIMLAR

2.1. HUKUK ALANINDA YAPAY ZEKA UYGULAMALARI

Hukuk, doğası gereği devasa miktarda yapılandırılmamış metin verisi (kanunlar, içtihatlar, yönetmelikler, akademik makaleler) içerir. Yapay zekâ teknolojileri, bu veri yığını içerisinden anlamlı bilgilerin çıkarılması, analiz edilmesi ve sentezlenmesi süreçlerini otomatikleştirerek hukuk profesyonellerine önemli bir verimlilik artışı sunmaktadır [4]. Literatürdeki çalışmalar genellikle iki ana paradigma etrafında toplanmıştır: Hukuki Sonuç Tahmini ve Hukuki Bilgi Erişimi.

2.1.1. Hukuki Sonuç Tahmini

Bu yaklaşım, geçmiş dava verilerini kullanarak gelecekteki davaların sonuçlarını (kabul, ret, kısmen kabul vb.) tahmin etmeyi amaçlar. Metin sınıflandırma ve istatistiksel modelleme temel yöntemlerdir. Katz vd. tarafından yapılan çalışmada, ABD Yüksek Mahkemesi kararlarının verileri kullanılarak, SVM gibi klasik makine öğrenmesi algoritmalarıyla dava sonuçları yüksek bir doğrulukla tahmin edilmiştir [1].

Son yıllarda geliştirilen "PILOT" gibi yeni yaklaşımlar, içtihat sisteminin zaman içerisindeki evrimini ve emsal karar yapısını da dikkate alarak daha sofistike tahminler ortaya koymaktadır [11]. Ayrıca, derin öğrenme temelli modellerin (örneğin BERT, BigBird) bazı yargı alanlarında insan uzmanları geçtiği gösterilmiştir [12]. Yine de, açıklanabilirlik, güvenilirlik ve şeffaflık açısından bu sistemlerin hâlen sınırlılıkları bulunmaktadır [13].

2.1.2. Hukuki Bilgi Erişimi

Bu yaklaşımın amacı, geniş bir hukuk metin külliyatı içerisinden kullanıcı sorgularına uygun belgeleri bulmak ve sunmaktır. Bu alandaki geleneksel sistemler, Westlaw ve LexisNexis gibi ticari platformlar olup, anahtar kelime tabanlı arama yöntemleriyle (TF-IDF, BM25) çalışmaktadır [2]. Ancak bu yöntemler, metnin anlamsal derinliğini veya kavramsal ilişkileri yeterince yakalayamaz.

Bu eksiklik, günümüzde Retrieval-Augmented Generation (RAG) sistemlerinin gelişmesine yol açmıştır. RAG yapıları, önce bilgiye dayalı belgeleri geri getirir (retrieval), ardından bu belgelerle büyük dil modellerini besleyerek daha güvenilir ve bağlamsal yanıtlar üretir [7]. Bu sayede, hem halüsinasyon riski azaltılır hem de sonuçların doğruluğu artar [14]. Ayrıca, hukuk özelinde geliştirilen LegalBench-RAG gibi benchmark'lar, bu sistemlerin doğruluğunu değerlendirmede önemli rol oynamaktadır [15].

2.2. BÜYÜK DİL MODELLERİ VE RETRIEVAL-AUGMENTED GENERATION (RAG)

2.2.1. Transformer Mimarisi ve Büyük Dil Modelleri (LLM)

Modern doğal dil işleme alanının temelini, 2017 yılında tanıtılan Transformer mimarisi ve bu mimarideki "self-attention" mekanizması oluşturur [5]. Bu mimari, kelimeler arasındaki uzun mesafeli ilişkileri yüksek başarıyla modelleyerek LLM'lerin (GPT, LLaMA vb.) geliştirilmesini mümkün kılmıştır.

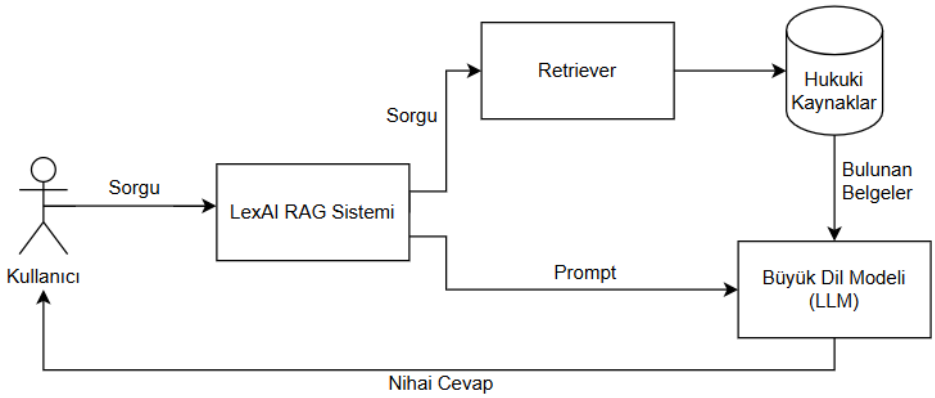
Bu modeller, özetleme, metin üretme ve analiz gibi görevlerde insan üstü başarılar sergilese de temel bir sınırlamaya sahiptir: Eğitildikleri veriyle sınırlı olduklarından güncel bilgiye ulaşamazlar, özel belgeleri bilemezler ve bazen kaynak göstermeden yanlış bilgi üretebilirler [6, 7]. Bu nedenle, LLM'ler RAG yapılarıyla birleştirilerek hem bilgiye dayalı hem de bağlamsal doğru yanıtlar üretmesi sağlanır [14, 15].

2.2.2. Retrieval-Augmented Generation (RAG) Mimarisi

RAG, LLM'lerin yukarıda belirtilen zayıflıklarını gidermek için tasarlanmış bir mimaridir. Bir LLM'in bir soruya doğrudan cevap vermesi yerine, RAG süreci iki ana aşamadan oluşur:

1. Retrieval (Bilgi Çekme): Kullanıcı sorgusu, harici bir bilgi bankasında (bizim projemizde hukuki kaynaklar) arama yapmak için kullanılır. En alakalı metin parçaları (chunk'lar) bulunur.
2. Generation (Cevap Üretme): Bulunan bu metin parçaları, orijinal soru ile birlikte bir prompt (talimat metni) içinde LLM'e sunulur. LLM, cevabını kendi hafızasından değil, sadece bu kendisine sunulan güncel ve güvenilir bağlamdan yola çıkarak üretir [6].

Retrieval-Augmented Generation (RAG) mimarisi ve temel çalışma akışı Şekil 2.1'de gösterilmiştir.



Şekil 2.1: Retrieval-Augmented Generation (RAG) Mimarisi ve Temel Çalışma Akışı

2.3. GELİŞMİŞ RETRIEVAL TEKNİKLERİ

Bu çalışmada, standart bir RAG sisteminden daha ileri gidilerek, retrieval aşamasının doğruluğunu artırmak için birden fazla gelişmiş teknik bir arada kullanılmıştır.

2.3.1. Metin Vektörleştirme (Embeddings)

Anlamsal aramanın temelinde, metinlerin anlamını sayısal vektörler şeklinde temsil eden "embedding" yöntemleri yer alır. Bu çalışmada, çok dilli destek sunan ve yüksek performansıyla öne çıkan “intfloat/multilingual-e5-large” modeli tercih edilmiştir [16].

Bu tür modeller, klasik anahtar kelime eşlemesine kıyasla bağlamsal anlam benzerliğini çok daha başarılı bir şekilde yakalayabilir [8].

2.3.2. Hibrit Arama (Hybrid Search)

Bu çalışmada, tek bir arama yönteminin sınırlılıklarını aşmak ve daha isabetli sonuçlar elde etmek amacıyla hibrit bir arama mimarisi benimsenmiştir. Bu yaklaşım, anlamsal benzerliği yakalayan Yoğun Arama (Dense Retrieval) ile anahtar kelime eşleşmelerinde güçlü olan Seyrek Arama’nın (Sparse Retrieval) avantajlarını bir araya getirir [7, 17]. Bu sayede hem bağlam hem de kelime eşleşmesi dikkate alınarak daha dengeli ve etkili bir bilgi erişimi sağlanır.

2.3.2.1. Yoğun Vektör Araması (Dense Vector Retrieval)

Yoğun aramanın temel amacı, sorgunun harfi harfine aynısını değil, anlamını ve niyetini anlayan ve bu anlama en yakın metin parçalarını getirmektir. Bu, "avukatın sorumluluğu" gibi bir sorgu için, içinde "avukat" kelimesi geçmese bile "vekâlet görevinin özenle yerine getirilmesi" gibi bir konuyu içeren metinleri bulabilmesini sağlar.

Bilgi erişim sisteminin ilk aşaması, metin verisinin anlamsal olarak işlenebilir sayısal temsillere dönüştürülmesidir. Bu süreçte, bilgi tabanında yer alan her bir metin parçası (chunk), intfloat/multilingual-e5-large gibi modern bir dil modeli kullanılarak işlenmiştir. Bu model, her metin parçasını, içeriğindeki anlamsal özü temsil eden yüksek boyutlu bir vektöre (embedding) dönüştürür [16]. Bu vektör temsilleri, metinlerin anlamsal yakınlıklarını hesaplamak için bir temel oluşturur ve metnin salt kelime içeriğinin ötesinde, bağlamsal anlamını da yakalar.

Milyonlarca metinden üretilen bu vektörler arasından verimli bir şekilde arama yapabilmek, sistemin performansı için kritik bir öneme sahiptir. Bu zorluğun üstesinden gelmek amacıyla, Meta AI tarafından geliştirilen yüksek performanslı FAISS (Facebook AI Similarity Search) kütüphanesinden yararlanılmıştır. FAISS, ham kuvvet (brute-force) yaklaşımıyla her vektörü birbiriyle tek tek karşılaştırmak yerine, vektör uzayını daha küçük alt bölgelere ayıran ve aramayı yalnızca sorgu vektörünün bulunduğu en olası bölgelerle sınırlayan optimize edilmiş indeksleme yapıları kullanır [10]. Bu yaklaşım, arama süresini önemli ölçüde azaltarak büyük ölçekli veritabanlarında dahi anlık sonuçlar elde etmeyi mümkün kılar.

Vektörler arasındaki anlamsal benzerlik düzeyini nicel olarak ifade etmek için bir metrik kullanılması gerekmektedir. Bu çalışmada, metrik olarak L2 Mesafesi (Öklid Mesafesi) tercih edilmiştir. Vektör uzayındaki iki vektör arasındaki doğrudan geometrik mesafeyi

ölçen L2 Mesafesi, anlamsal ilişki için sezgisel bir ölçüt sunar. Bu metrikte, iki vektör arasındaki mesafenin daha düşük olması, temsil ettikleri metinler arasındaki anlamsal benzerliğin daha yüksek olduğu anlamına gelmektedir. FAISS gibi kütüphaneler, bu tür mesafe metriklerini kullanarak en yakın komşuları (nearest neighbors) hızla bulmak üzere optimize edilmiştir [18].

2.3.2.2. Seyrek Vektör Araması (Sparse Vector Retrieval)

Seyrek aramanın temel amacı, sorgudaki anahtar kelimelerin metinlerdeki önemine ve sıklığına dayalı olarak en alakalı belgeleri bulmaktır. Bu yöntem, özellikle spesifik hukuki terimler, kanun maddesi numaraları veya kişi/kurum isimleri içeren sorgularda yüksek isabet sağlar.

Bu çalışmada, endüstri standardı olan Okapi BM25 algoritması kullanılmıştır. BM25, geleneksel TF-IDF (Term Frequency-Inverse Document Frequency) modelinin geliştirilmiş bir versiyonudur ve skor hesaplamasında üç ana bileşeni dikkate alır:

- Term Frequency (TF): Sorgudaki kelimenin belgede ne kadar sık geçtiği,
- Inverse Document Frequency (IDF): Kelimenin tüm belge havuzunda ne kadar nadir olduğu (örn: "ihtiyati tedbir" gibi nadir terimler daha fazla ağırlık taşır),
- Belge Uzunluğu Normalizasyonu: Daha uzun belgelerin fazla anahtar kelime içerme olasılığına karşı normalizasyon yaparak, kısa belgelerin haksız şekilde düşük skor almasını önler [17].

2.3.2.3. Sonuçların Birleştirilmesi: Ağırlıklı Puan Birleştirme (Weighted Score Fusion

Hibrit aramanın son aşaması, iki sistemden gelen sonuçların akıllıca birleştirilmesidir. FAISS'ten gelen mesafe skorları (küçük = iyi) ile BM25'ten gelen alaka skorları (büyük = iyi) farklı ölçeklerde olduğundan doğrudan toplanamazlar. Bu nedenle şu adımlar uygulanır:

Normalizasyon: Her iki arama motorundan gelen skorlar, karşılaştırılabilir hale gelmeleri için önce Min-Max Normalizasyonu gibi bir yöntemle ortak bir ölçeğe (genellikle 0 ile 1 arası) getirilir. Bu aşamada, FAISS'in mesafe skorları, 1 / (1 + mesafe) gibi bir formülle benzerlik skoruna dönüştürülür, böylece her iki metrikte de yüksek skor daha iyi anlamına gelir.

Ağırlıklı Birleştirme: Normalize edilmiş skorlar, her arama yöntemine verilen ağırlık katsayısı α\alphaα ile çarpılarak birleştirilir:

$$\text{HybridSkor} = (\alpha \cdot \text{NormalizeEdilmisDenseSkor}) + ((1-\alpha) \cdot \text{NormalizeEdilmisSparseSkor})$$

Bu çalışmada, anlamsal aramanın biraz daha baskın olması için α değeri 0.6-0.7 aralığında ayarlanmıştır.

Nihai Sıralama: Hesaplanan HybridSkor'a göre aday belgeler yeniden sıralanır. Böylece hem anlamsal olarak zengin hem de anahtar kelime açısından isabetli sonuçlar en üstte yer alır ve sonraki aşama olan Yeniden Sıralama (Re-ranking) katmanına girdi olur [18].

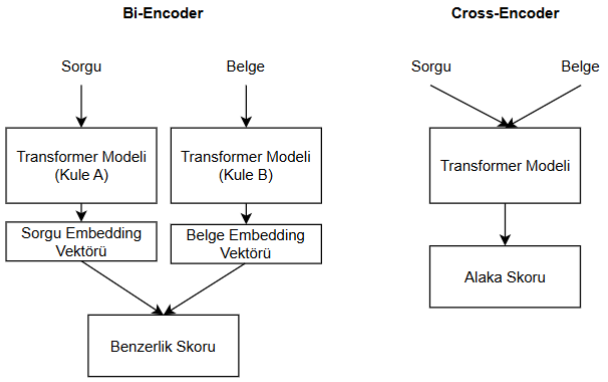
2.3.3. Yeniden Sıralama (Re-ranking) ve Cross-Encoder'lar

Hibrit aramadan gelen aday listesindeki en doğru sonuçları en üste taşımak için bir yeniden sıralama katmanı kullanılmıştır. Bu işlem, cross-encoder/mmarco-mMiniLMv2-L12-H384-v1 gibi bir Cross-Encoder modeli ile yapılmıştır. Bi-Encoder'ların aksine Cross-Encoder'lar, (sorgu, belge) çiftini aynı anda işleyerek çok daha hassas bir alaka skoru üretir ve arama kalitesini önemli ölçüde artırır. Bu katman, RAG sisteminin "precision" (isabet) oranını maksimize etmek için kritik bir rol oynar [19].

Bilgi erişim (information retrieval) sistemlerinde kullanılan temel model mimarileri, işlevsellik ve verimlilik açısından farklılaşan iki ana kategoriye ayrılmaktadır. Bu mimariler, Bi-Encoder ve Cross-Encoder olarak adlandırılır ve genellikle birbirini tamamlayan rollerde kullanılırlar [8, 19].

İlk temel mimari olan Bi-Encoder, sorgu (query) ve belgelerin (documents) birbirinden bağımsız olarak işlendiği bir yapı sunar. Bu çalışmada kullanılan intfloat/multilingual-e5-large modeli gibi Bi-Encoder'lar, biri sorgu diğeri ise belgeler için olmak üzere iki ayrı sinir ağı "kulesi" (tower) kullanır. Her bir kule, kendisine verilen metni işleyerek sabit boyutlu bir vektör (embedding) üretir. Arama işlemi sırasında, önceden hesaplanıp FAISS gibi bir vektör veritabanında indekslenmiş olan milyonlarca belge vektörü arasından, gelen sorgu için anlık olarak hesaplanan vektöre en yakın olanlar bulunur. Bu yakınlık, genellikle kosinüs benzerliği veya L2 mesafesi gibi metrikler kullanılarak belirlenir. Bu mimarinin en büyük avantajı, belge vektörlerinin önceden hesaplanabilmesi sayesinde arama işleminin son derece hızlı olmasıdır. Buna karşın en belirgin dezavantajı, sorgu ve belge arasındaki etkileşimin, yalnızca son aşamada gerçekleşen vektör karşılaştırması ile sınırlı kalmasıdır. Bu durum, metinler arasındaki derin anlamsal ilişkilerin ve ince nüansların gözden kaçırılmasına neden olabilir.

İkinci temel mimari olan Cross-Encoder ise sorgu ile belge arasındaki ilişkiyi daha derinlemesine modellemek üzere tasarlanmıştır. Bu çalışmanın yeniden sıralama (re-ranking) aşamasında kullanılan cross-encoder/mmarco-mMiniLMv2-L12-H384-v1 modeli, bu mimariye bir örnektir. Bi-Encoder'ların aksine Cross-Encoder'lar, sorgu ve belge metinlerini [CLS] sorgu metni [SEP] belge metni [SEP] gibi tek bir birleşik girdi halinde alır. Bu birleşik metin, Transformer modelinin tüm katmanlarından birlikte geçerek işlenir. Bu yaklaşımın en önemli avantajı, modelin sahip olduğu "cross-attention" mekanizmasının, sorgudaki her bir kelimenin belgedeki her bir kelimeyle doğrudan etkileşime girmesine imkân tanımasıdır. Bu sayede model, sorgu ve belge arasındaki alaka düzeyini çok daha bağlamsal ve hassas bir seviyede değerlendirerek isabetli bir alaka skoru üretebilir. Ancak bu mimarinin en büyük dezavantajı, her bir (sorgu, belge) çifti için modelin baştan sona çalıştırılmasını gerektirmesi nedeniyle son derece yavaş olmasıdır. Bu işlem maliyeti, Cross-Encoder'ları milyonlarca belgeden oluşan geniş koleksiyonlarda ilk aday tespiti için pratik olmayan bir seçenek haline getirir ve genellikle Bi-Encoder ile daraltılmış daha küçük aday kümelerini yeniden sıralamak amacıyla kullanılmalarına yol açar.Bi-Encoder ve Cross-Encoder mimarilerinin karşılaştırmalı gösterimi Şekil 2.2'de sunulmuştur [8, 19].



Şekil 2.2 : Bi-Encoder ve Cross-Encoder Mimarilerinin Karşılaştırmalı Gösterimi

Cross-Encoder'ların yavaşlığı ve Bi-Encoder'ların hızından kaynaklanan bu değiş tokuş (trade-off), bu çalışmada kullanılan iki aşamalı modern retrieval mimarisini zorunlu kılar [19]:

1. Aşama: Aday Tespiti (Candidate Retrieval): Hızlı olan Bi-Encoder (e5-large) ve Hibrit Arama (FAISS + BM25) yöntemleri kullanılarak, kullanıcı sorgusuyla ilgili olabilecek potansiyel olarak geniş bir aday listesi (örneğin en iyi 25-50 belge) hızla bulunur. Bu aşama, "recall" (yakalama) odaklıdır.

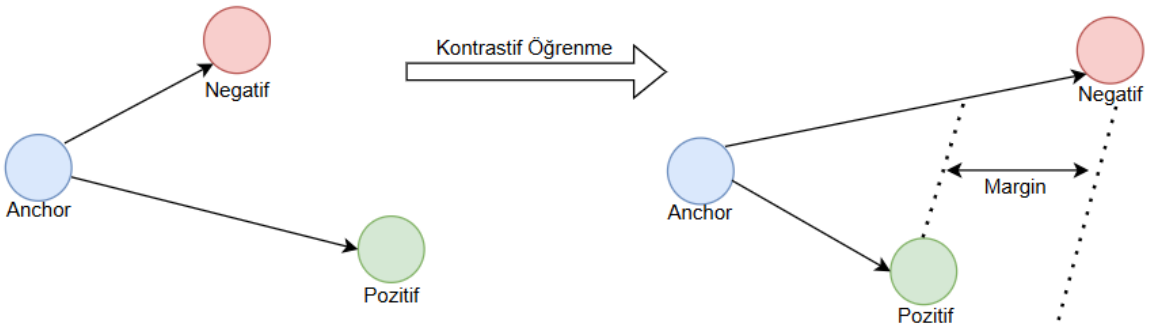
2. Aşama: Hassas Sıralama (Precision Re-ranking): Yavaş ama çok daha isabetli olan Cross-Encoder modeli, sadece bu küçük aday listesindeki belgeleri alır. Her bir adayı orijinal sorguyla karşılaştırarak hassas bir alaka skoru üretir. Bu skorlara göre adaylar yeniden sıralanır ve en yüksek skora sahip olanlar (örneğin ilk 3-5 belge) nihai bağlam olarak seçilir. Bu aşama, "precision" (isabet) odaklıdır.

Bu iki aşamalı yapı, hem hızlı bir sistem performansı sunar hem de son kullanıcıya sunulan bilginin maksimum alaka düzeyine sahip olmasını sağlar.

2.4. DİL MODELLERİNİN ALANA ÖZGÜ UZMANLAŞTIRILMASI

2.4.1. Embedding Modelleri için Fine-tuning (İnce Ayar) ve Kontrastif Öğrenme

Genel amaçlı embedding modellerinin, spesifik bir alandaki (örn: Türk Hukuku) anlamsal nüansları tam olarak yakalayamaması bilinen bir sorundur. Bu sorunu aşmak için, bu çalışmada kullanılan e5-large modeli, alana özgü bir veri seti ile yeniden eğitilmiştir. Bu uzmanlaştırma süreci, Kontrastif Öğrenme (Contrastive Learning) adı verilen bir yaklaşıma dayanır. Kontrastif öğrenmenin amacı, modele benzer örnekleri birbirine yakınlaştırmayı, benzemeyen örnekleri ise birbirinden uzaklaştırmayı öğretmektir. Kontrastif Öğrenme süreci Şekil 2.3'te görselleştirilmiştir [9].



Şekil 2.3 : Kontrastif Öğrenme

Bu öğrenme sürecini yönetmek için, bu çalışmada Triplet (Üçlü) Veri Seti formatı kullanılmıştır. Her bir veri örneği, üç bileşenden oluşan bir üçlüdür:

Anchor (Çapa): Referans olarak alınan ana metin veya sorgu.
Positive (Pozitif): Anchor ile anlamsal olarak çok benzer veya doğrudan ilişkili olan metin.
Negative (Negatif): Anchor ile anlamsal olarak ilişkisiz veya yanıltıcı şekilde benzer olan metin.

Eğitimin hedefi, embedding uzayında (vektör uzayında) Anchor'ın Pozitif'e olan mesafesini, Anchor'ın Negatif'e olan mesafesinden daha küçük hale getirmektir. Bu, Triplet Loss (Üçlü Kayıp) adı verilen özel bir kayıp fonksiyonu ile sağlanır.

Triplet Loss'un temel matematiksel formülasyonu şöyledir:

$$L(A,P,N)=max(0,d(A,P)-d(A,N)+\alpha)$$

Burada:

- A, P, N: Sırasıyla Anchor, Pozitif ve Negatif örneklerin vektörleridir.
- d(A, P): Anchor ve Pozitif vektörleri arasındaki mesafeyi (örn: Öklid mesafesi) ifade eder.
- d(A, N): Anchor ve Negatif vektörleri arasındaki mesafeyi ifade eder.

- α (alpha): Marjin (margin) olarak bilinen pozitif bir değerdir.

Bu fonksiyonun amacı, pozitif çiftin mesafesinin (d(A, P)), negatif çiftin mesafesinden (d(A, N)) en az marjin (alpha) kadar daha küçük olmasını sağlamaktır. Yani, *d(A, P) + alpha < d(A, N)* koşulunu hedefler.

Eğer bu koşul sağlanıyorsa, model doğru bir ayrım yapmış demektir. Formüldeki d(A, P) - d(A, N) + alpha ifadesi negatif bir değer alır ve max(0, ...) fonksiyonu sayesinde kayıp (loss) 0 olur. Modelin bu örnek için ağırlıklarını güncellemesine gerek kalmaz.
Eğer bu koşul sağlanmıyorsa, modelin bir hata yaptığı anlaşılır. d(A, P) - d(A, N) + alpha ifadesi pozitif bir değer alır ve bu değer, modelin kaybını oluşturur. Eğitim algoritması (backpropagation), bu kaybı minimize etmek için modelin ağırlıklarını günceller. Bu güncelleme, P vektörünü A'ya yaklaştırır ve/veya N vektörünü A'dan uzaklaştırır.

Bu çalışmada kullanılan MultipleNegativesRankingLoss ise, bu temel Triplet Loss mantığının daha verimli ve gelişmiş bir versiyonudur. Her bir eğitim adımında, bir (anchor, positive) çifti için, o anki mini-batch (küçük veri partisi) içindeki diğer tüm pozitif örnekleri "zor negatif" olarak ele alır. Bu, her adımda çok sayıda üçlü oluşturarak modelin daha hızlı ve daha etkili bir şekilde öğrenmesini sağlar [9].

2.4.2. Prompt Mühendisliği (Prompt Engineering)

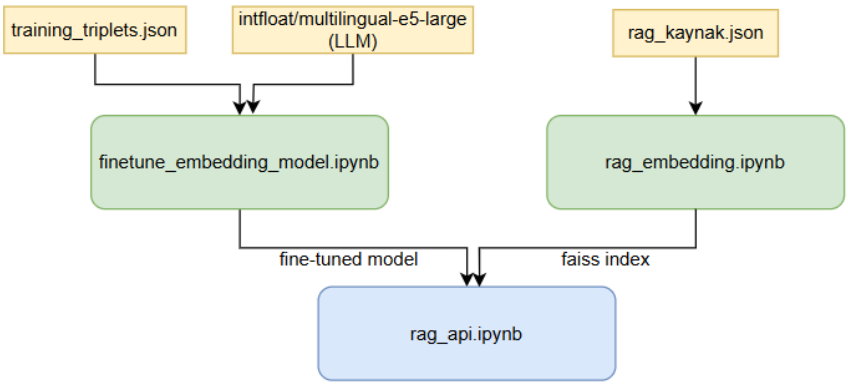
Cevap üretme aşamasında kullanılan Llama 3.1 8B-Instruct modelinin davranışını kontrol etmek ve istenen formatta cevaplar üretmesini sağlamak için detaylı bir Sistem Prompt'u tasarlanmıştır. Bu prompt, modele bir persona ("Bilişim Hukuku Uzmanı"), uyması gereken kurallar (kaynak gösterme, tavsiye vermeme) ve cevaplarını sunması gereken bir şablon (Analiz, Sonuç, Dayanak) atayarak, üretilen metnin kalitesini ve tutarlılığını artırmıştır [3, 6].

3. KULLANILAN ARAÇ VE YÖNTEM

Bu çalışmada, Türk Bilişim Hukuku alanında uzmanlaşmış, gelişmiş bir yapay zeka destekli soru-cevap asistanı olan LexAI'nin tasarımı ve gerçeklenmesi amaçlanmıştır. Sistemin geliştirilmesinde, en güncel doğal dil işleme teknikleri ve Retrieval-Augmented Generation (RAG) mimarisi temel alınmıştır [6,7,15]. Bu bölümde, projenin mimarisi, veri işleme adımları, kullanılan algoritmalar ve geliştirilen modeller detaylı bir şekilde açıklanacaktır.

3.1. PROJE MİMARİSİ VE VERİ AKIŞI

Proje, her biri belirli bir amaca hizmet eden üç ana modülden oluşmaktadır: Veri Hazırlama ve İndeksleme, Embedding Modelinin İnce Ayarı (Fine-tuning) ve RAG API Servisi. Bu modüler yapı, sürecin yönetilebilirliğini ve tekrarlanabilirliğini artırmaktadır. Veri akışı, yapılandırılmamış hukuki metinlerin toplanmasıyla başlar, bu metinlerin işlenip arama indekslerinin ve özel modellerin oluşturulmasıyla devam eder ve son olarak bu varlıkların bir API aracılığıyla sunulmasıyla son bulur. Projenin genel sistem mimarisi ve RAG API geliştirme sürecinin akış şeması Şekil 3.1'de gösterilmiştir.



Şekil 3.1 : Genel Sistem Mimarisi ve RAG API Geliştirme Sürecinin Akış Şeması

3.2. BİLGİ TABANININ OLUŞTURULMASI VE VERİ İŞLEME

Sistemin cevaplarının doğruluğu ve kapsamı, temel aldığı bilgi tabanının kalitesine doğrudan bağlıdır. Bu amaçla, bilişim hukukuyla ilgili geniş bir külliyat oluşturulmuştur.

3.2.1. Veri Kaynakları ve Yapılandırma

Bilgi tabanı; TCK, KVKK, 5651 Sayılı Kanun, EHK, ETK gibi temel kanunların yanı sıra, bu kanunların uygulanmasına yönelik yönetmelikler, tebliğler, güncel KVKK Kurul Kararlarını içermektedir. Bu dokümanlar, yapılarına göre iki farklı strateji ile işlenmiştir:

1. Kanunlar: Madde ve fıkra yapısı korunarak, her bir fıkra bir child chunk ve ait olduğu madde ise parent document olarak yapılandırılmıştır.
2. Uzun Metinler (Kararlar, Yönetmelikler vb.): Metnin tamamı bir parent document olarak alınmış ve anlamsal bütünlüğü koruyan, paragraflar arası 1-2 cümlelik çakışma (overlap) içeren daha küçük child chunk'lara bölünmüştür.

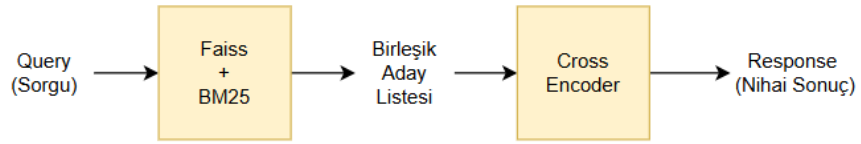
Bu veriler, rag_kaynak.json (child'lar) ve parents.json (parent'lar) olmak üzere iki ana dosyada normalize edilerek saklanmıştır.

3.2.2. Veri İşleme Araçları

Veri okuma, işleme ve yapılandırma adımlarında Python programlama dili ve pandas, numpy, json gibi standart veri işleme kütüphanelerinden yararlanılmıştır.

3.3. GELİŞMİŞ RETRIEVAL KATMANI: HİBRİT ARAMA VE YENİDEN SIRALAMA

Kullanıcı sorgusuna en alakalı bilgiyi yüksek isabetle getirmek için, literatürdeki en iyi pratikleri bir araya getiren çok aşamalı bir retrieval katmanı tasarlanmıştır. Gelişmiş Retrieval Katmanı'nın çok aşamalı yapısı Şekil 3.2'de gösterilmektedir.



Şekil 3.2 : İki Aşamalı Gelişmiş Bilgi Erişim (Retrieval) Mimarisi

3.3.1. Yoğun Vektör İndeksleme (FAISS)

Anlamsal arama için, sentence-transformers kütüphanesi ve intfloat/multilingual-e5-large modeli kullanılarak, rag_kaynak.json dosyasındaki tüm child chunk'ların metinleri (başlıkları ile birlikte) 1024 boyutlu vektörlere dönüştürülmüştür. Bu vektörler üzerinde yüksek hızlı benzerlik araması yapabilmek için Meta AI'ın FAISS kütüphanesi ile bir IndexFlatL2 indeksi oluşturulmuştur. Bu indeks, sorgu vektörüne Öklid mesafesi en yakın olan metin parçalarını bulur.

3.3.2. Seyrek Vektör İndeksleme (BM25)

Anahtar kelime eşleşmelerini yakalamak amacıyla, aynı metinler üzerinde rank_bm25 kütüphanesi kullanılarak bir Okapi BM25 indeksi kurulmuştur. Bu indeks, sorgudaki terimlerin metinlerdeki frekansına ve nadirliğine dayalı bir skor üreterek, özellikle teknik terimler ve madde numaraları içeren sorgularda yüksek isabet sağlar.

3.3.3. Hibrit Arama ve Cross-Encoder ile Yeniden Sıralama

Retrieval süreci şu şekilde işler:

1. Bir sorgu geldiğinde, hem FAISS hem de BM25 indekslerinde paralel bir arama yapılır ve her iki sistemden de en iyi N adet aday sonuç alınır.
2. Bu iki sonuç kümesi birleştirilir ve skorları normalize edilerek bir hibrit skor hesaplanır.
3. Bu hibrit aday listesi, daha sonra cross-encoder/mmarco-mMiniLMv2-L12-H384-v1 modeline dayalı bir Cross-Encoder'a gönderilir. Cross-Encoder, her bir adayı sorguyla birlikte işleyerek çok daha hassas bir alaka skoru üretir.

4. Son olarak, Cross-Encoder tarafından en yüksek skoru alan ilk K adet child chunk'ın ait olduğu parent document'lar, LLM'e bağlam olarak sunulmak üzere seçilir.

3.4. EMBEDDING MODELİNİN İNCE AYARI VE DEĞERLENDİRİLMESİ

Genel amaçlı embedding modelinin Türk Bilişim Hukuku alanındaki performansını artırmak amacıyla bir ince ayar (fine-tuning) süreci yürütülmüştür.

3.4.1. Veri Seti Hazırlığı

"Soru Üret & Zor Negatif Bul" stratejisiyle yaklaşık 3000 adet yüksek kaliteli (anchor, positive, negative) üçlüsü içeren training_triplets.json dosyası oluşturulmuştur. Bu süreçte Llama 3.1 modeli soru üretmek (anchor), mevcut RAG sistemi ise yanıltıcı ama benzer negative örnekleri bulmak için kullanılmıştır.

3.4.2. Modelin Eğitilmesi ve Değerlendirilmesi

sentence-transformers kütüphanesi kullanılarak, e5-large modeli, oluşturulan bu üçlü veri seti ve MultipleNegativesRankingLoss kayıp fonksiyonu ile 1 epoch boyunca eğitilmiştir. Eğitimin başarısını ölçmek amacıyla, manuel olarak oluşturulan evaluation_set.json test seti kullanılmıştır.

3.5. ÜRETKEN KATMAN VE API SERVİSİ

3.5.1. Cevap Üretimi ve Prompt Mühendisliği

Sistemin son katmanı, retrieval sonucunda elde edilen bağlama dayalı olarak nihai cevabı üreten meta-llama/Meta-Llama-3.1-8B-Instruct modelidir. LLM'lerin doğası gereği yaratıcı ve bazen kontrolsüz olabilen yapısını, projenin gerektirdiği hassas ve güvenilir hukuki analiz formatına dönüştürmek için Gelişmiş Prompt Mühendisliği (Advanced Prompt Engineering) teknikleri uygulanmıştır.

Bu süreç, modele sadece bir soru sormaktan öte, ona bir kimlik atayan, görevini adım adım tarif eden, uyması gereken katı kurallar koyan ve çıktısını belirli bir şablona zorlayan bir Sistem Prompt'u (System Prompt) tasarlamayı içerir. Bu, modelin tutarlı, doğru, kaynak gösteren ve hukuki tavsiye vermekten kaçınan bir "Bilişim Hukuku Uzmanı" personasında cevap vermesini sağlayarak, üretilen metnin kalitesini ve güvenilirliğini en üst düzeye çıkarmıştır.

Bu prompt, LLM davranışını hassas bir şekilde kontrol etmek amacıyla, birbiriyle entegre çalışan birkaç temel bileşen üzerine inşa edilmiştir. İlk olarak, prompt'un başında modele bir Persona ve Uzmanlık Alanı (PERSONA / ROL) atanır [20]. Bu talimat, modelin genel bilgi havuzundan ziyade, "Türkiye Bilişim Hukuku Uzmanı" kimliğiyle ilişkili dil kalıplarını, terminolojiyi ve argüman yapılarını aktive etmesini teşvik ederken, uzmanlık alanının sınırlarının çizilmesiyle de modelin alakasız konularda yorum yapmasını engeller. Ardından, modelin görevi ve izlemesi gereken mantıksal akış, "Chain of Thought" (Düşünce Zinciri) olarak bilinen bir teknikle tanımlanır. İŞLEM ADIMLARI, modelin cevabı doğrudan üretmesi yerine, adım adım düşünmesini (önce anla, sonra alaka kontrolü yap, sonra formüle et) talep ederek, üretilen cevabın mantıksal tutarlılığını ve gerekçelendirme kalitesini önemli ölçüde artırır [21]. Modelin çıktısının tutarlılığını ve profesyonelliğini

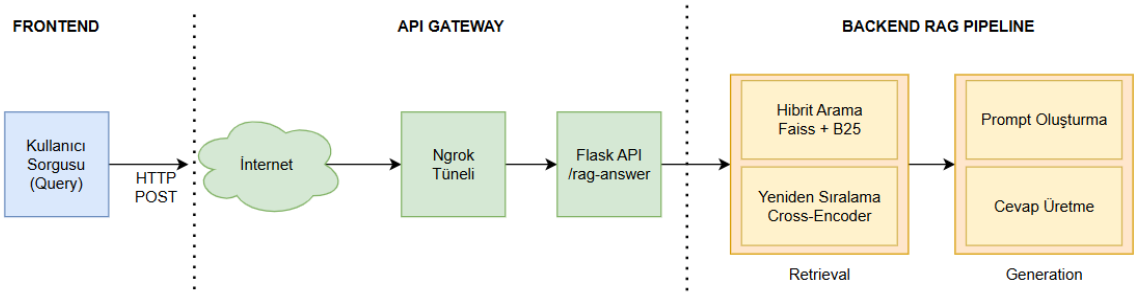
garanti altına almak için ise, serbest metin yerine Analiz:, Sonuç: ve Dayanak: başlıklarını içeren katı bir Cevap Şablonu'na uyması zorunlu kılınmıştır. Bu yapı, kullanıcı için okunabilirliği artırırken, Dayanak bölümü RAG mimarisinin temel amacı olan şeffaflığı ve kaynak gösterme yeteneğini somutlaştırır. Mimari'nin en kritik katmanını ise, sistemin güvenilirliğini sağlayan Güvenlik Mekanizmaları ve Kısıtlamalar (Guardrails) oluşturur. Bu kurallar, retrieval katmanının sunduğu bağlam yetersiz veya alakasız olduğunda modelin bilgi uydurmasını (halüsinasyon) kesin bir dille yasaklar, cevaplarını sadece ve sadece sunulan bağlama dayandırmaya zorlar ve sistemi hukuki tavsiye vermekten alıkoyarak onu bir "bilgilendirme aracı" olarak doğru bir şekilde konumlandırır [22].

3.5.2. API Servisi

Tüm bu RAG akışı, Python tabanlı Flask web framework'ü ile bir RESTful API (/rag-answer endpoint'i) olarak paketlenmiştir. Bu API, sunum ve test aşamasında Ngrok aracı kullanılarak internet üzerinden erişilebilir hale getirilmiş ve React ile geliştirilen kullanıcı arayüzüyle entegre edilmiştir.

3.6. SİMÜLASYON ORTAMI VE KULLANILAN TEKNOLOJİLER

Bu bölümde, LexAI projesinin geliştirilmesi, eğitilmesi ve çalıştırılması süreçlerinde kullanılan programlama dilleri, platformlar ve kütüphaneler detaylı bir şekilde açıklanacaktır. Proje, temel olarak yapay zeka ve makine öğrenmesi modellerini barındıran bir Python backend'i ile kullanıcı etkileşimi için geliştirilmiş bir React frontend'inden oluşmaktadır. LexAI sisteminin katmanlı mimarisi ve uçtan uca veri akışı Şekil 3.3'te gösterilmiştir.



Şekil 3.3 : LexAI Sisteminin Katmanlı Mimarisi ve Uçtan Uca Veri Akışı

3.6.1. Geliştirme Ortamı ve Programlama Dilleri

Python: Tüm backend işlemleri, veri işleme, model eğitimi ve API geliştirme süreçleri için ana programlama dili olarak Python (sürüm 3.10+) kullanılmıştır. Python'un, yapay zeka ve veri bilimi alanındaki zengin kütüphane ekosistemi, projenin hızlı ve verimli bir şekilde geliştirilmesini sağlamıştır.
Google Colab: Projenin geliştirme, eğitim ve sunum aşamalarında ana platform olarak Google Colab'dan yararlanılmıştır. Colab'ın sunduğu NVIDIA A100 gibi yüksek performanslı GPU'lara erişim imkanı, özellikle Büyük Dil Modellerinin (LLM) yüklenmesi ve embedding modelinin ince ayar (fine-tuning) gibi yoğun hesaplama gerektiren işlemlerin makul sürelerde tamamlanması için kritik bir rol oynamıştır.

JavaScript/TypeScript ve React: Kullanıcı arayüzü (frontend), modern ve bileşen tabanlı bir web uygulaması oluşturmak amacıyla React kütüphanesi kullanılarak geliştirilmiştir. Kodun tip güvenliğini ve yönetilebilirliğini artırmak için TypeScript tercih edilmiştir.

3.6.2. Yapay Zeka ve Veri İşleme Kütüphaneleri

PyTorch: Tüm derin öğrenme modellerinin temelinde yer alan, esnek ve güçlü bir bilimsel hesaplama kütüphanesidir. Modellerin hem eğitimi hem de çalıştırılması (inference) bu framework üzerinde gerçekleştirilmiştir.
Hugging Face Transformers: Son teknoloji Transformer tabanlı yapay zeka modellerine erişim için endüstri standardı haline gelmiş bir kütüphanedir. Bu proje kapsamında meta-llama/Meta-Llama-3.1-8B-Instruct gibi üretken dil modellerinin ve cross-encoder modellerinin yüklenmesi ve kullanılması bu kütüphane aracılığıyla yapılmıştır.
Sentence-Transformers: Metin embedding'i oluşturma, fine-tuning ve retrieval performansını değerlendirme gibi görevleri büyük ölçüde basitleştiren, Transformers üzerine inşa edilmiş bir kütüphanedir. intfloat/multilingual-e5-large modelinin yüklenmesi, MultipleNegativesRankingLoss ile eğitilmesi ve InformationRetrievalEvaluator ile test edilmesi bu kütüphane sayesinde mümkün olmuştur.
FAISS (Facebook AI Similarity Search): Anlamsal arama katmanının bel kemiğidir. Milyonlarca vektör arasında yüksek hızda benzerlik araması yapabilen, Meta AI tarafından geliştirilmiş bir kütüphanedir.
rank_bm25: Geleneksel ancak etkili bir anahtar kelime tabanlı arama algoritması olan Okapi BM25'i uygulamak için kullanılmıştır. Hibrit arama altyapısının seyrek vektör (sparse) bacağını oluşturur.

3.6.3. API Geliştirme ve Sunma

Flask: Geliştirilen RAG sistemini dış dünyaya bir servis olarak sunmak için kullanılan, Python tabanlı, hafif ve minimalist bir web framework'üdür. /rag-answer gibi bir RESTful API endpoint'i oluşturmak için tercih edilmiştir.
Ngrok: Colab üzerinde yerel olarak çalışan Flask sunucusuna, internet üzerinden erişilebilen genel bir URL (public URL) atamak için kullanılan bir tünelleme (tunneling) servisidir. Bu araç, React ile geliştirilen frontend uygulamasının, coğrafi olarak farklı bir yerde çalışan backend API'si ile iletişim kurmasını sağlamıştır.
Axios: React tabanlı frontend uygulamasından, Flask ile oluşturulan backend API'sine asenkron HTTP POST istekleri göndermek için kullanılmıştır.

4. SİSTEMİN GERÇEKLENMESİ

4.1. VERİ İŞLEME VE INDEKSLEME (RAG_EMBEDDING.IPYNB)

Sistemin bilgi tabanını oluşturan hukuki külliyatın, yapay zeka modelleri tarafından işlenebilir ve aranabilir hale getirilmesi bu aşamanın temel amacıdır.

Kaynakların Toplanması ve Yapılandırılması: Projenin bilgi tabanı olarak belirlenen kanunlar, yönetmelikler, tebliğler ve kurul kararları, rag_kaynak.json ve parents.json dosyalarında, Bölüm 3'te detaylandırılan Parent Document Retriever mimarisine uygun olarak yapılandırılmıştır. Aranabilir birimler olan child chunk'lar ve tam bağlamı sunan parent document'lar bu adımda ayrıştırılmıştır.
İndekslerin Oluşturulması: Gelişmiş bir arama performansı için hibrit bir yaklaşım benimsenmiştir.
FAISS İndeksi: rag_kaynak.json dosyasındaki tüm child chunk'lar, intfloat/multilingual-e5-large modeli kullanılarak 1024 boyutlu vektörlere dönüştürülmüş ve bu vektörler üzerinde hızlı anlamsal arama yapabilmek için bir FAISS IndexFlatL2 indeksi oluşturulmuştur.
BM25 İndeksi: Aynı metinler, anahtar kelime tabanlı arama için rank_bm25 kütüphanesi kullanılarak bir Okapi BM25 indeksine dönüştürülmüştür.
Bu sürecin sonunda, API'nin kullanacağı my_faiss_index.index ve bm25_index.pkl dosyaları oluşturulup kalıcı olarak kaydedilmiştir.

4.2. EMBEDDING MODELİNİN İNCE AYARI VE DEĞERLENDİRİLMESİ (FINETUNE_EMBEDDING_MODEL.IPYNB)

Bu aşamanın amacı, genel amaçlı embedding modelini Türk Bilişim Hukuku alanında uzmanlaştırmak ve bu uzmanlaşmanın getirdiği performans artışını sayısal olarak kanıtlamaktır.

4.2.1. Modelin Eğitilmesi

"Soru Üret & Zor Negatif Bul" stratejisiyle oluşturulan training_triplets.json veri seti kullanılarak, intfloat/multilingual-e5-large modeli üzerinde ince ayar (fine-tuning) yapılmıştır. Eğitim, sentence-transformers kütüphanesi ve MultipleNegativesRankingLoss kayıp fonksiyonu ile 1 epoch boyunca gerçekleştirilmiştir. Eğitim sonucunda, modelin Türk hukuki metinleri arasındaki anlamsal ilişkileri daha iyi kavrayan özel bir versiyonu elde edilmiş ve my_finetuned_embedding_model adıyla kaydedilmiştir.

4.2.2. Deneysel Sonuçlar ve Bulgular

Fine-tuning işleminin başarısını ölçmek için, eğitimde kullanılmayan evaluation_set.json veri seti ile bir karşılaştırma testi yapılmıştır. Orijinal model ile fine-tune edilmiş modelin performansı, standart Bilgi Erişim (Information Retrieval) metrikleri olan Recall@K ve Mean Reciprocal Rank (MRR) kullanılarak değerlendirilmiştir.

Tablo 4.1 : Orijinal ve Fine-tuned Modellerin Performans Karşılaştırması

	Orijinal Model	Fine-tuned Model	İyileşme
Recall@1	0.728814	0.830508	0.101695
Recall@3	0.864407	0.949153	0.084746
Recall@5	0.966102	0.949153	-0.016949
Recall@10	0.983051	0.949153	-033898
MRR@10	0.821186	0.881356	0.060169

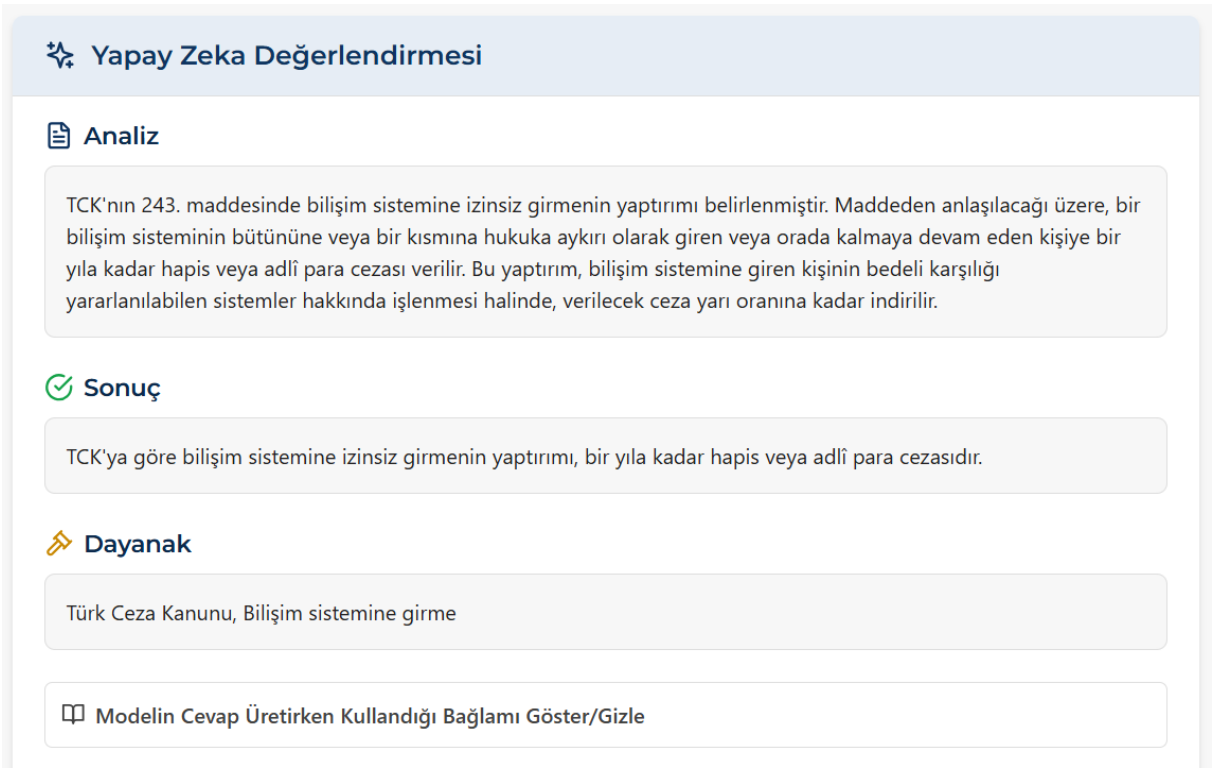
Tablo 4.1'den de görüleceği üzere, gerçekleştirilen fine-tuning işlemi, modelin doğru cevabı ilk sırada bulma başarısını (Recall@1) %10'un üzerinde artırmıştır. Özellikle doğru cevabın sıralamadaki yerini ölçen MRR metriğindeki artış, yeni modelin sadece doğru sonuçları bulmakla kalmayıp, bu sonuçları daha üst sıralara taşıdığını da göstermektedir. Bu bulgular, alana özgü fine-tuning işleminin, RAG sistemlerinin retrieval performansını artırmada ne kadar etkili bir yöntem olduğunu kanıtlamaktadır.

4.3. RAG API'SİNİN GERÇEKLENMESİ VE SİSTEM DEMONSTRASYONU (RAG_API.IPYNB)

Son aşamada, projenin önceki adımlarında hazırlanan tüm bileşenler bir araya getirilerek, canlı bir soru-cevap servisi olarak çalışacak bir web apı'si geliştirilmiştir. Bu apı'nin altyapısı, python tabanlı, hafif bir web framework'ü olan flask kullanılarak kurulmuştur. Geliştirme ve sunum aşamalarinda, colab üzerinde çalişan bu yerel sunucunun internet üzerinden erişilebilir hale gelmesi ise ngrok servisi ile sağlanmiştir.

Sistemin çalışma mantığı, kullanıcıdan bir istek geldiğinde tüm rag akışının tetiklenmesi üzerine kuruludur. Apı, bir sorgu aldığında, daha önce eğitilmiş olan fine-tuned embedding modelini, cross-encoder modelini, faıss ve bm25 indekslerini ve cevap üretimi için llama 3.1 8b-ınstruct dil modelini entegre bir şekilde kullanır. Örneğin, şekil 4.1'de gösterildiği gibi, bir kullanıcının arayüze "TCK'ya göre bilişim sistemine izinsiz girmenin yaptırımı nedir?" şeklinde bir hukuki sorun girmesiyle süreç başlar. Bu sorgu üzerine sistem, ilgili kanun, yönetmelik ve ilke kararlarından oluşan bilgi tabanını kullanarak, bölüm 3'te detaylandırılan rag pipeline'ını işletir ve kullanıcıya gerekçeli, kaynak gösteren bir cevap döndürür. Sistemin üretmiş olduğu yapılandırılmış cevap (analiz, sonuç ve dayanak) şekil 4.2'de görülmektedir. Modelin cevap üretirken kullandiği dayanak bölümü, şekil 4.3'teki gibi, ilgili kanun maddelerini ve başliklarini içermektedir.

Şekil 4.1 : LexAI Kullanıcı Arayüzü ve Örnek Sorgu Girişi



Şekil 4.2 : Sistemin Ürettiği Yapılandırılmış Cevap (Analiz, Sonuç ve Dayanak)

Şekil 4.3 : Retrieval Sonucunda LLM'e Sunulan Bağlam Metinleri ("Dayanak" Bölümü)

5. TARTIŞMA VE SONUÇ

5.1. ÇALIŞMANIN ÖZETİ VE ELDE EDİLEN BAŞARI

Bu çalışmanın temel amacı, Büyük Dil Modelleri'nin (LLM) doğruluk ve güvenilirlik sorunlarını, Retrieval-Augmented Generation (RAG) mimarisi ile aşmaktı. Bu doğrultuda, TCK, KVKK, 5651 Sayılı Kanun ve ilgili ikincil mevzuat ile kurul kararlarını içeren kapsamlı bir bilgi tabanı oluşturulmuştur [23,24,25,26,27,28,29,30] . Sistemin bilgiye erişim doğruluğunu maksimize etmek için, anlamsal arama yapan FAISS ile anahtar kelime tabanlı arama yapan BM25'i birleştiren bir Hibrit Arama mekanizması kurulmuştur. Bu mekanizmadan gelen aday sonuçlar ise, daha hassas bir alaka analizi için Cross-Encoder tabanlı bir Yeniden Sıralama (Re-ranking) katmanından geçirilmiştir.

Projenin en önemli katkısı, genel amaçlı intfloat/multilingual-e5-large embedding modelinin, Türk hukuki metinlerinden yarı otomatik olarak oluşturulan bir veri seti ile ince ayar (fine-tuning) yapılarak uzmanlaştırılmasıdır. Yapılan deneysel değerlendirmeler, fine-tune edilmiş modelin, orijinal modele kıyasla doğru belgeyi ilk sırada bulma başarısını (Recall@1) ve genel sıralama kalitesini (MRR) ölçülebilir şekilde artırdığını kanıtlamıştır. Son olarak, bu gelişmiş retrieval katmanından gelen bağlam, Llama 3.1 8B-Instruct modeline, özel olarak tasarlanmış bir sistem prompt'u ile sunulmuş ve modelin kaynak göstererek, gerekçeli ve tutarlı cevaplar üretmesi sağlanmıştır.

5.2. SONUÇLARIN YORUMLANMASI VE KARŞILAŞILAN ZORLUKLAR

Elde edilen sonuçlar, alana özgü fine-tuning işleminin, RAG sistemlerinin isabet oranını artırmada ne kadar kritik bir rol oynadığını göstermektedir. Genel amaçlı bir modelin dahi yüksek bir başlangıç performansına sahip olmasına rağmen, hukukun kendine özgü terminolojisi ve anlamsal nüansları için yapılan özel bir eğitimin, "zor" sorgularda doğru cevabı bulma yeteneğini belirgin şekilde iyileştirdiği görülmüştür.

Proje sürecinde karşılaşılan en büyük zorluk, fine-tuning işlemi için gerekli olan yüksek kaliteli eğitim veri setinin hazırlanması olmuştur. "Soru Üret & Zor Negatif Bul" stratejisi bu süreci otomatize etse de, üretilen verinin kalitesini sağlamak için insan denetimi ve veri temizliği önemli bir zaman ve efor gerektirmiştir. Bir diğer zorluk ise, LLM'in davranışlarını kontrol etmektir. İlk denemelerde modelin, yetersiz bağlam verildiğinde halüsinasyon görme veya kendini tekrar etme eğiliminde olduğu gözlemlenmiştir. Bu sorun, "Kritik Kurallar ve Kısıtlamalar" içeren detaylı bir prompt mühendisliği süreci ile büyük ölçüde aşılmıştır.

5.3. GELECEK ÇALIŞMALAR VE POTANSİYEL UYGULAMA ALANLARI

Bu çalışma, gelecekteki birçok araştırmaya ve uygulamaya öncü olabilir. Sistemin geliştirilebileceği bazı alanlar şunlardır:

LLM Fine-tuning: Bu çalışmada embedding modeli eğitilmiştir. Bir sonraki adım olarak, cevap üretme modeli olan Llama 3.1'in de, hukuki dil stiline, argüman kurma biçimine ve kaynak gösterme formatına daha iyi uyması için (prompt, ideal_cevap) çiftleri ile fine-tune edilmesi, sistemin ürettiği metinlerin kalitesini daha da artıracaktır.

Bilgi Tabanını Genişletme: Sistemin bilgi tabanı, daha fazla Yargıtay, Danıştay ve Anayasa Mahkemesi kararı ile zenginleştirilebilir. Özellikle farklı hukuk dallarını (örn: İş Hukuku, Ticaret Hukuku) kapsayacak şekilde genişletilebilir.
Diyalog Yönetimi: Mevcut sistem her soruyu bağımsız olarak ele almaktadır. Sisteme bir hafıza ve diyalog yönetimi yeteneği eklenerek, kullanıcıların takip soruları sorabildiği daha akıcı bir "sohbet" deneyimi yaratılabilir.

Bu teknolojinin potansiyel uygulama alanları oldukça geniştir:

Hukuk Büroları: Avukatlar için bir ön araştırma asistanı olarak, dava dosyaları ile ilgili mevzuatı ve emsal kararları hızla bulabilir.
Hukuk Fakülteleri: Öğrenciler için, kanun maddeleri ve pratik uygulamaları arasındaki bağlantıyı interaktif bir şekilde öğreten bir eğitim aracı olabilir.
Şirketlerin Hukuk ve Uyum Departmanları: KVKK veya e-ticaret gibi alanlardaki uyum süreçlerinde hızlı bir başvuru kaynağı olarak kullanılabilir.

5.4. NİHAİ SONUÇ

Bu tez çalışması, modern yapay zeka tekniklerinin, özellikle de alana özgü olarak uzmanlaştırılmış Retrieval-Augmented Generation sistemlerinin, hukuk gibi karmaşık ve metin-yoğun bir alanda nasıl başarılı bir şekilde uygulanabileceğini göstermiştir. Geliştirilen LexAI sistemi, hibrit arama, yeniden sıralama ve fine-tuning gibi gelişmiş yöntemleri bir araya getirerek, sadece bilgi bulmanın ötesinde, bu bilgiyi analiz edip sunabilen yenilikçi bir karar destek aracıdır. Bu çalışma, hukuki bilgiye erişimi demokratikleştirme ve araştırma süreçlerini daha verimli hale getirme yolunda atılmış önemli bir adımı temsil etmektedir.

KAYNAKLAR

[1] Katz, D. M., Bommarito, M. J., & Blackman, J. (2017). A general approach for predicting the behavior of the Supreme Court of the United States. PloS one, 12(4), e0174698.

[2] Thomson Reuters. (2023). Westlaw Edge AI. Product Page. / LexisNexis. (2023). Lexis+ AI. Product Page.

[3] Allen & Overy. (2023, February). Allen & Overy partners with Harvey to bring generative AI to the legal industry. Press Release.

[4] Birtane, Ş. (2024). Hâkime Yardımcı Yapay Zekâ. Türkiye Adalet Akademisi Dergisi, (59), 235-266.

[5] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I. (2017). Attention Is All You Need. arXiv preprint arXiv:1706.03762.

[6] Lewis, P., et al. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. arXiv preprint arXiv:2005.11401.

[7] Gao, Y., et al. (2023). Retrieval-Augmented Generation for Large Language Models: A Survey. arXiv preprint arXiv:2312.10997.

[8] Reimers, N., Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. arXiv preprint arXiv:1908.10084.

[9] Wang, L., et al. (2022). Text Embeddings by Weakly-Supervised Contrastive Pre-training. arXiv preprint arXiv:2212.03533.

[10] Johnson, J., Douze, M., Jégou, H. (2017). Billion-scale similarity search with GPUs. arXiv preprint arXiv:1702.08734.

[11] Cao, L., et al. (2024). PILOT: Modeling Precedent Evolution in Legal Outcome Prediction. arXiv preprint arXiv:2401.06752.

[12] Zhong, H., et al. (2020). Legal Judgment Prediction via Topological Learning. In ACL.

[13] Aletras, N., et al. (2016). Predicting judicial decisions of the European Court of Human Rights: A Natural Language Processing perspective. PeerJ Computer Science.

[14] Tang, R., et al. (2023). Context-Aware RAG: Enhancing Factuality and Retrieval Alignment. arXiv preprint arXiv:2305.13245.

[15] LegalBench Team. (2023). LegalBench-RAG: A Benchmark for Legal Retrieval-Augmented Generation. Retrieved from legalbench.org.

[16] Wang, K., Li, J., & Xiong, C. (2022). Text Embedding Models in Information Retrieval: A Survey. arXiv preprint arXiv:2210.00241.

[17] Hofstätter, S., Möller, M., & Bevendorff, J. (2022). A Survey on Hybrid Search: When Sparse and Dense Retrieval Unite. *arXiv preprint arXiv:2206.07694*.

[18] Guo, M., Zhang, Z., Fan, K., Sun, L., & Lin, J. (2023). Efficient and Accurate Vector Search via Distance-aware Product Quantization. *arXiv preprint arXiv:2303.12922*.

[19] Thakur, N., Yates, A., & Cohan, A. (2021). BEIR: A Heterogeneous Benchmark for Zero-shot Evaluation of Information Retrieval Models. arXiv preprint arXiv:2104.08663.

[20] Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C.L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., Lowe, R. (2022). Training language models to follow instructions with human feedback. arXiv preprint arXiv:2203.02155.

[21] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., Zhou, D. (2022). Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. arXiv preprint arXiv:2201.11903.

[22] Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y., Chen, D., Dai, W., Chan, H.S., Madotto, A., Fung, P. (2022). Survey of Hallucination in Natural Language Generation. arXiv preprint arXiv:2202.03629.

[23] 5237 Sayılı Türk Ceza Kanunu, Mevzuat Bilgi Sistemi. https://www.mevzuat.gov.tr/mevzuat?MevzuatNo=5237&MevzuatTur=1&MevzuatTertip=5.

[24] 6698 Sayılı Kişisel Verilerin Korunması Kanunu, Mevzuat Bilgi Sistemi. https://www.mevzuat.gov.tr/mevzuat?MevzuatNo=6698&MevzuatTur=1&MevzuatTertip=5.

[25] 5651 Sayılı İnternet Ortamında Yapılan Yayınların Düzenlenmesi ve Bu Yayınlar Yoluyla İşlenen Suçlarla Mücadele Edilmesi Hakkında Kanun, Mevzuat Bilgi Sistemi. https://mevzuat.gov.tr/mevzuat?MevzuatNo=5651&MevzuatTur=1&MevzuatTertip=5.

[26] Kurul Kararları, KVKK. Erişim Tarihi: 18 Mayıs 2025 https://kvkk.gov.tr/Icerik/5419/Kurul-Kararlari.

[27] Yönetmelikler, KVKK. Erişim Tarihi: 22 Mayıs 2025 https://kvkk.gov.tr/Icerik/5257/Yonetmelikler.

[28] Tebliğler, KVKK. Erişim Tarihi: 13 Mayıs 2025 https://kvkk.gov.tr/Icerik/5258/Tebligler.

[29] İlke Kararları, KVKK. Erişim Tarihi: 2 Mayıs 2025 https://kvkk.gov.tr/Icerik/7201/Ilke-Kararlari.

[30] Kurul Karar Özetleri, KVKK. Erişim Tarihi: 1 Haziran 2025 https://kvkk.gov.tr/Icerik/5406/Kurul-Karar-Ozetleri.

EKLER

ÖZGEÇMİŞ

Musa Berke ŞENGÖZ

2001 İstanbul doğumlu. İlkokulu Söğütlüçeşme İlkokulunda tamamladı. Ortaokulu Söğütlüçeşme Ortaokulunda, liseyi de Ataköy Anadolu Lisesi'nde tamamladı. 2020 yılından beri İstanbul Üniversitesi Cerrahpaşa Mühendislik Fakültesi Bilgisayar Mühendisliği bölümüne devam etmektedir.

Yağmur GÖKÇEKLİ

2001 Zonguldak doğumlu. İlkokulu Riyaziyeci Salih Zeki İlkokulunda tamamladı. Ortaokulu Yavuz Selim Ortaokulunda, liseyi Beşiktaş Anadolu Lisesi'nde tamamladı. 2020 yılından beri İstanbul Üniversitesi Cerrahpaşa Mühendislik Fakültesi Bilgisayar Mühendisliği bölümüne devam etmektedir.