

Webinar Transkripti: E-Ticarette Öneri Sistemleri Mimarisi

Konuşmacı: Furkan Aksu (Machine Learning Engineer)

Moderatörler: Enes Fehmi Manan, Selin Çıldam

[Bölüm 1: Two-Tower Mimarisi ve Eğitim Mantığı]

Furkan Aksu:

...(Sunumun başı eksik, model mimarisinden devam ediyor)...

Araya Transformers katmanları da konulabilir ama temel mantık genelde "Neural Network" (Sinir Ağları) üzerinden ilerliyor. Item (Ürün) ve User (Kullanıcı) bilgileri giriyor, işleniyor ve sonunda bunların **Dot Product**'ı (Nokta Çarpımı) alınıyor. Buradaki amaç, bu Dot Product sonucunu maksimize etmek. Yani bir **Loss Function** (Kayıp Fonksiyonu) kuruyoruz.

Target (Hedef) değişkenin nasıl bulunduğunu anlatayım: Diyelim ki ben Item A'ya tıkladım, sonra Item B'ye tıkladım, sonra Item C'ye tıkladım. Bu aslında A, B ve C arasında bir ilişki var demektir. Yani A'ya ve B'ye tıklayan kullanıcı, C'ye de tıklayabilir anlamına geliyor. Elimde böyle milyonlarca veri var.

Burada **Pozitif Target (1)** ve **Negatif Target (0)** oluşturmamız gerekiyor.

- **Pozitifler:** Zaten elimde olan, kullanıcının gerçekten tıkladığı silsiledir (A -> B).
- **Negatifler:** İlgisi olmayan şeylerdir. Bunu sağlamak için "Random" (Rastgele) ürünler getiriyoruz. Örneğin A'ya ve B'ye tıkladı ama Z ürünüyle alakası yok. Z'yi negatif örnek olarak seçiyoruz.

Sonra **Binary Cross Entropy** kullanarak bu hatayı minimize ediyoruz. Bunu milyonlarca adımla ve milyonlarca kullanıcıyla yaptığımız zaman, her User ve her Item için bir **Embedding** (Vektör temsili) elde edebiliyoruz. Artık kullanıcının ilgi duyduğu veya duyabileceği her şey, onun rakamlarında gizli hale geliyor.

[Bölüm 2: Aday Belirleme (Retrieval) ve Sıralama (Ranking)]

Furkan Aksu:

Genel bir özet geçeyim: Elimde User Feature'lar ve Item Feature'lar var. Bunları ikiye ayırıp Neural Network'e sokuyorum, Embedding'ler elde ediyorum. Loss Function ile minimize edip, Dot Product'ı maksimize ediyorum.

Sonuçta elimde bir kullanıcı için vektör oluştu. Gidiyorum veritabanına, "Elimde 2 milyon ürün var, bu kullanıcıya en yakın 1000 tanesini getir" diyorum (Retrieval aşaması).

Şimdi elimde 2000 tane aday ürün var. Amacım kullanıcıya 20 tane öneri sunmak. Bunun için **Ranking Modelleri** devreye giriyor. Bu 2000 adayı alıp, örneğin CTR'ı (Tıklama Oranı) veya Similarity'i (Benzerlik) maksimize edecek şekilde sıralıyorum.

Bu modeller (Learning to Rank) temel olarak üçe ayrılıyor:

1. **Pointwise:** Klasik yöntemler.

2. **Pairwise:** En yaygın kullanılanı. (Örn: LambdaMART, XGBoost). Mantığı şudur: "A ve B ürünlerinin yerini değiştirsem listenin toplam kalitesi artar mı?"
3. **Listwise:** Daha kompleks liste bazlı yöntemler.

[Bölüm 3: İş Kuralları (Re-Ranking) ve Kişiselleştirme]

Furkan Aksu:

Tüm modellerin güzel çalışması için güzel feature'lar (özellikler) gerekir.

- **User Features:** Kullanıcının yaşı, ekran saati, cihaz tipi, ziyaret ettiği videolar, ülkesi vb.
- **Item Features:** Ürünün kendi kategorisinde ne kadar popüler olduğu, markasındaki popülerliği, geçmiş satış datası vb.

Son olarak **Business Rules** (İş Kuralları) devreye girer. Şirkete özel kurallar uygularız:

- "Arka arkaya aynı kategoriden ürün gösterme."
- "Stokta olmayanları ele."
- "Yeni ürünleri öne çıkar."

[Bölüm 4: Performans Ölçümü (Offline & Online)]

Furkan Aksu:

Performansı iki aşamada ölçeriz:

1. **Offline Metrics (Canlıya almadan önce):**
 - **Precision@K:** Önerdiğim ürünler ne kadar doğru?
 - **NDCG:** Kullanıcının ilgilendiği ürünü 1. sırada mı gösterdim, 4. sırada mı? (Sıralama kalitesi).
 - *Not:* Test setini zaman serisine göre ayırırız. İlk 27 gün Train, son 3 gün Test gibi. Geleceği görmemeli (Data Leakage olmamalı).
2. **Online Metrics (A/B Testi):**
 - Kullanıcıları ikiye böleriz. Yarıları eski modeli, yarıları yeni modeli görür.
 - **CTR (Click Through Rate)** veya **Conversion Rate (Dönüşüm Oranı)** gibi metriklerde istatistiksel olarak anlamlı bir fark var mı diye bakarız.
 - Bazen basit bir Lojistik Regresyon, karmaşık bir Two-Tower modelden daha iyi CTR getirebilir. Bunu A/B testi yapmadan bilemeyiz.

[Bölüm 5: Teknoloji Yığını (Tech Stack)]

Furkan Aksu:

Hangi teknolojileri kullanıyoruz?

- **Veri Ambarı: BigQuery.** Milyonlarca satır veriyi (30-40 milyon müşteri, 30 milyon ürün) hızlıca işleyebiliyor. SQL tabanlı.
- **Programlama: Python** (Scikit-learn, PyTorch).
- **Canlı Servis (Serving):** Canlıda öneriyi çok hızlı (milisaniyeler içinde) sunmak için **NoSQL (Redis/Couchbase vb.)** kullanıyoruz. Ayrıca Item vektörlerini tutmak için **Vector Database**'ler kullanılıyor.
- **Orkestrasyon: Apache Airflow.** Modelin her hafta güncellenmesi, pipeline'ların otomatik çalışması için.
- **Dağıtım: Docker** (Image oluşturmak için) ve **Kubernetes** (Canlıda servis etmek/scale etmek için).

[Bölüm 6: Karşılaşılan Zorluklar (Challenges)]

Furkan Aksu:

1. **Cold Start (Soğuk Başlangıç):** Yeni kullanıcının veya yeni ürünün geçmiş verisi yoktur.
 - **Çözüm:** Yeni kullanıcılara "Popüler Ürünler" gösteriyoruz. Yeni ürünler içinse ürünün isminden/fotoğraftan Embedding çıkaran modeller kullanıp benzerlerini bulmaya çalışıyoruz.
2. **Sparsity (Seyreklik):** 30 milyon kullanıcı var ama her kullanıcının etkileşimi sadece 3-5 ürünle. Matris çok boş. Two-Tower mimarisi bunu çözmede başarılı.
3. **Feedback Loops (Yankı Odası):** Model popüler ürünü önerir, kullanıcı tıklar, ürün daha popüler olur. Yeni ürünler arada kaynar.
 - **Çözüm:** Yeni eklenen ürünlere yapay skor eklemeleri (boosting) veya keşif (exploration) stratejileri uyguluyoruz.

[Soru - Cevap Bölümü]

Enes Fehmi Manan: Genel olarak mimari akışını (Pipeline) özetleyebilir miyiz?

Furkan Aksu:

1. Elimde milyonlarca ürün var.
2. **Retrieval (Two-Tower):** Bunu 1000 adaya indiriyorum.
3. **Ranking (XGBoost/LGBM vb.):** Bu 1000 adayı puanlayıp sıralıyorum.
4. **Re-Ranking (Business Rules):** 30 tane ürünü kurallara göre (stok, kampanya, çeşitlilik) filtreliyorum.
5. Final liste kullanıcıya gidiyor.

Enes Fehmi Manan: Retrieval aşamasındaki (aday belirleme) başarıyı nasıl ölçüyorsunuz? Belki yanlış elemeyeyle doğru ürünleri başta atıyoruz?

Furkan Aksu:

Train ve Test seti ayırımımız var. Modelin getirdiği adaylar (Recall), son 3 günde kullanıcının gerçekten tıkladığı ürünleri kapsıyor mu diye bakıyoruz. Yani etiketli gerçek datayla karşılaştırıyoruz.

Enes Fehmi Manan: Modeli her hafta eğitiyorsanız, o hafta sisteme yeni giren ürünler ne oluyor? Önerilmiyor mu?

Furkan Aksu:

Evet, şu anki yapıda yeni giren ürünler bir sonraki eğitime kadar (Pazartesi'den Pazartesi'ye) tam performanslı önerilemiyor. Bunu aşmak için ürün isminden/kategorisinden benzerlik kuran yan yöntemler denedik ama şu anki production (canlı) yapımızda haftalık eğitim döngüsü var.

Chat Sorusu: LLM (Large Language Model) ile Fine-tuning yaparak öneri sistemi yapılamaz mı? (Örn: LLaMA vb.)

Furkan Aksu:

LLM'leri canlı öneri sisteminde kullanmak **maliyet ve hız** açısından şu an çok zor. Milyonlarca istek için LLM çağırmak çok pahalı ve yavaş (Latency sorunu). Ayrıca yapılandırılmış çıktı (Structured Output) almak zor. O yüzden **Open Source Embedding Modelleri** (örn: CLIP, multilingual-e5) kullanıp, veriyi vektöre çevirip Two-Tower gibi daha hafif mimarilerle çalışmak şu an daha mantıklı.

Selin Çıldam: Firma genç bir firma (4 yaşında) olmasına rağmen sadık bir kitlesi var, bu çalışmalar geliştikçe deneyim daha da iyileşecektir.

Katılımcı Sorusu: Üye olmayan kullanıcılar (Guest) için ne yapıyorsunuz?

Furkan Aksu:

Çerezler (Cookies) üzerinden bir "Guest ID" tutuyoruz. Kullanıcı üye olmasa bile o cihazdaki hareketlerini takip edip ona göre öneri sunuyoruz.

Katılımcı Sorusu: Similarity (Benzerlik) için Dot Product mı Cosine Similarity mi kullanıyorsunuz?

Furkan Aksu:

Dot Product kullanıyoruz. Dot Product'ta popüler ürünler (vektör büyüklüğü fazla olduğu için) biraz daha öne çıkabiliyor, bu bizim işimize geliyor. Cosine Similarity normalize ettiği için popülerliği biraz eziyor.

Kapanış:

(Enes Fehmi Manan): Ağzınıza sağlık, çok teknik ve ufuk açıcı bir oturum oldu. E-ticarete bu işlerin mutfağını görmek çok kıymetli.

(Furkan Aksu): Ben teşekkür ederim, herkese iyi akşamlar.