# Ürün Görsellerinin (RGB) Renk Kanalları ile K-Means Kümeleme

BLM 5110 Makine Öğrenmesi

Metin Uslu - 235B7014

Yapılan çalışmanın kodlarına ColorClustering{.ipynb, .py ve .html} dosyaları üzerinden erişebilirsiniz. Projenin tüm repository’sine aşağıdaki adres üzerinden erişim sağlayabilirsiniz. Aşağıda belirtilen açıklamaların çok daha fazlasına ColorClustering notebook dosyası üzerinden erişebilirsiniz.

GitHub: <https://github.com/metinuslu/blm5110_color_clustering>

1. **Özet: Yaptığınız çalışmayı, elde ettiğiniz sonuçları özet olarak veriniz.**

Bu çalışmada Roboflow platformundan alınan farklı çözünürlükte ve doğal ortamlarda çekilmiş, kırmızı, yeşil, mavi, beyaz ve gri renk çeşitliliğinde ürün görselleri R,G,B renk kanalları üzerinden kümelenerek renkler bazında kümelenmesi amaçlanmıştır. Öncesinde resimlerin array haline getirilmesi, ardından renk kanallarının orijinal ve normalize histogramlarının elde edilmesi ve görselleştirilmesi sağlanmıştır. Devamında kendimiz implemente ettiği K-Means algoritması kullanılarak her resim için R, G ve B normalize histogramları üzerinden kümele çalışması yapılmıştır. Küme merkezleri rastgele belirlenmesi ile initialize edilen algoritmanın %49 bir doğruluk performansı elde edilmiştir. Yine 10 farklı deneme ile elde edilen performans aralığı %32 ile %54 arasında olduğu gözlemlenmiştir.

1. **Giriş: Ödev konusunu tanıtan 1 paragraflık bir giriş yapınız. Bu çalışmanın nerelerde kullanılabileceğinden bahsediniz.**

Bu çalışma da Ürün görsellerinin R, G, B renk kanalları/özellikleri kullanılarak benzer renkte bulunan ürün görsellerinin kümelenmesi için yapılan bir çalışmasıdır. Bu çalışma aşağıdaki amaçlar için kullanılabilir;

1. Ürün görselleri üzerinde Renk özelliklerinin tespit edilmesinde kullanabilir.
2. Yine aynı zamanda ürünlerin renklerini kullanarak birbirlerine aynı ya da yakın renklere sahip ürünlerin tespit ederek bunlar gruplanabilir.
3. Bu aynı zamanda sonsuz sayıda bulunan ürün renk uzayını azaltmaya da (reduce) de katkı sağlayabilir. Mesela bazı renkleri (Bknz: Siyah, Gri, Kahverengi)koyu tonlar olarak, bazı renkleri (Bknz: Beyaz, Açık gri, Bej, Açık Sarı) açık tonlar, bazı renkleri(bordo, Koyu Kırmızı, Kırmızı, Turuncu) kızıl tonlar olarak indirgeyebilirler. Bu da Modelleme açısından bakıldığında “High Cardinality” problemine de çözüm sağlayacaktır.
4. **Sistem Tasarımı: Sisteminizin işlem adımlarını kısaca anlatınız.**
5. Resimlerin formatını hazır kütüphaneler kullanarak çözerek görüntüye ait matrisi elde ediniz. Bir görüntü piksellerin (R,G,B) bileşenlerinden oluştuğu bir matristir.

**create\_img\_to\_arr** fonksiyonu kullanılarak her bir resim dosyası array haline getirilmiştir. Burada resim dosyaları okunurken OpenCv library’si kullanılmıştır. OpenCv ile resimler varsayılan olarak BGR olarak okunmaktadır. Buna dikkat edilerek resim nesneleri BGR to RGB çevrilmiştir. Fonksiyon hem RGB hem de BGR resim arrayleri return etmektedir.

A screenshot of a computer

Description automatically generated

1. Resimleri oluşturan piksellerin (R,G,B) değerlerine göre her resmin renk histogramını (Her resimde R, G, B bileşenleri için ayrı ayrı olmak üzere toplam 3 histogram) elde ediniz. Histogram eldesi için de hazır kütüphane kullanabilirsiniz. Bir pikselin R,G,B bileşenlerinin değeri 0-255 arası değiştiği için her histogram dizisi 256 elemanlı olmalıdır.

**img\_to\_hist\_arr** fonksiyonu kullanılarak her bir resim nesnesinin R, G ve B histogramları OpenCv library’si kullanılarak elde edilmiştir. Fonksiyon R, G ve B için ayrı ayrı histogram arrayleri return etmektedir.

A screenshot of a computer

Description automatically generated

R, G ve B Histogramlarının görselleştirilmesi için **plot\_histogram** fonksiyonu yazılmıştır. Örnek birkaç resim için çıktıları aşağıda paylaşıyorum. ColorClustering Notebook (.ipynb, .html) üzerinden tamamına erişebilirsiniz.

A screenshot of a computer program

Description automatically generated

A group of graphs showing different colors

Description automatically generated

1. Her resimde, her renk bileşeni için histogram dizisindeki sonuçları resimdeki toplam piksel sayısına bölerek dizi elemanlarının değerini [0-1] aralığına normalize ediniz.

**img\_to\_norm\_hist\_arr** fonksiyonu kullanılarak R, G ve B histogramları 0 - 255 aralığından 0 - 1 aralığına normalize edilmiştir.

A screenshot of a computer

Description automatically generated

R, G ve B Histogram ve Normalize Histogram çıktılarını bir örnek resim üzerinden paylaşıyorum.

A screenshot of a computer screen

Description automatically generated

1. Resimlerin histogramlarını benzerliklerine göre k=5 değeri için(5 renk sınıfı olduğu için) k-means yöntemiyle kümeleyiniz. K-means’de başlangıç adımında rasgele seçeceğiniz k resmin histogramını k cluster’ın başlangıç merkezi olarak kullanınız.

KMeans class içerisine **fit**, **predict**, uzaklık hesaplama yöntemlerinin (Euclidean ve Manhattan) tercihi için **calculate\_distance** ve centroid değerlerinin rastgele belirlenmesi için **initialize\_centroids** fonksiyonları yazılmıştır.

A screenshot of a computer program

Description automatically generated

A white rectangular object with blue lines

Description automatically generated

1. **Deneysel Sonuçlar: Sistem başarısını değerlendirmek için aşağıdaki işlemleri yapınız.**

Model performasının ölçülmesi için her bir küme için {0..4} her kümede baskın olan Ürün Rengi o kümenin doğru sonucu olarak kabul edilip, elimizde bulunan bildiğimiz (ground truth) gerçek Ürün Renkleri ile Kümelerdeki Ürün Renklerinin karşılaştırılması ile Accuracy hesaplanmıştır. Yine öncesinde kümelerdeki Cluster x Count dağılımları ve ClusterId ve Color x Count dağılımları ColorClustering notebook dosyasında paylaşılmıştır.

* 1. Küme merkezlerini rasgele belirleyerek her k değeri için kümeleme işlemini 10 defa tekrarlayınız. Her kümeleme sonunda, kümelerde doğru cluster’da olan resim yüzdesini hesaplayınız.

Burada Cluster performansının ölçülmesi için **cluster\_evaluate** isimli bir fonksiyon yazılarak kümeleme işleminin performansı ölçülmüştür. 10 tekrar için ayrı ayrı Confusion Matrix sonuçlarına ColorClustering notebook dosyası üzerinden erişebilirsiniz.

A white background with text

Description automatically generated

A screenshot of a computer

Description automatically generated

A close-up of a computer screen

Description automatically generated

* 1. Hazırlayacağınız karışıklık matrisi (confusion matrix) üzerinde sonucu gösteriniz.

Burada Modeli oluşturduğumuz ve sadece 1 tekrar elde ettiğimiz Model Performansı ve Confusion Matrix yer almaktadır. 10 tekrar ile elde ettiğimiz Confusion Matrix’ler notebook içerisinden erişebilirsiniz.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

* 1. Her sınıf için doğru cluster’a yerleşmiş 5 örnek ve yanlış cluster’da bulunan 1 örnek resmi veriniz.

**Doğru Cluster Edilmiş 5 Örnek**

A white background with colorful text

Description automatically generated

A collage of clothes

Description automatically generated

A collage of clothes

Description automatically generated

A group of men's sweatshirts

Description automatically generated

**Yanlış Cluster Edilmiş 5 Örnek**



A collage of different coats

Description automatically generatedA screenshot of a computer screen

Description automatically generated

1. **Sonuç: Sizce kümeleme işlemi başarılı oldu mu? Yanlış kümelerde olan resimler sizce neden yanlış kümelerde yer aldı? Sistemin genel başarısını yorumlayınız. Başarısızlık sebebi olduğunu düşündüğünüz problemlerin giderilmesi için varsa önerilerinizi belirtiniz.**

Bu çalışmada Kümeleme performansı oldukça düşük bir başarıma sahip olduğunu tekrarlar ile görülmüştür. Hem tekli hem de 10 tekrarlı k adet farklı merkez noktası ile kümeleme algoritmasını (Accuracy) performansını ölçtüğümüz de performans maksimum %54’e ulaştığı gözlemlenmiştir. Bu da modelin performansının iyi olmadığını göstermektedir. Başarısızlığın sebepleri olarak;

1. Farklı Çözünürlük

* Çözünürlükler ön işleme ile belirli bir x,y çözünürlüğüne çekilebilir.

1. Doğal Ortam

* Tüm resimlerde doğal ortamların farklı olması, noiselere sebebiyet vermektedir. Bu sebeple eğer sağlanabiliyor ise aynı şartlar altında doğal ortamlarda resimlerin alınması.

1. Alt ve Üst kıyafetlerin birlikte bulunması sebebiyle birden fazla renk içerebiliyor olması

* Modelleme öncesi ön işleme adımı olarak semantic segmentasyon ürünler birbirlerinden ayrıştırılabilir.

1. Bazı ürünler için mankenli görsellerinin (yüz, boyun, el, vs.) bulunması resim üzerinde gürültüye(noise) sebep vermektedir

* Face detection ile mankenli görsellerin exclude edilmesi ya da detect edilen bölgenin croplanması oradan alınması yapılabilir.

1. Işık ve Diğer Etkenler

* Mümkünse tüm etkenlerin stabilizasyonunu sağlanması ya da tüm resimlerde aynı ön işleme teknikleri kullanılarak resimler de aynı standartizenin oluşmasının (variance indirgenmesi) sağlanması sağlanması

1. Daha Fazla Veri yada Veri Çeşitliğinin Artırılması
   * Eğer mümkünse daha fazla veri ile modellemenin tekrar edilmesi,
   * Eğer mümkün değil ise Data Augmentation ile her bir renk için ürün görsel çeşitliliğinin artırılması