KÜTAHYA SAĞLIK BİLİMLERİ ÜNİVERSİTESİ

[5]

Kalabalık Sayımı / Tespiti

Baki Dinç

14/06/2024

İçindekiler

1 GİRİŞ

Kalabalık sayımı (Crowd counting), büyük kalabalıkların sayısının veya yoğunluğunun tahmin edilmesiyle ilgili bir çalışmadır. İncelenen çalışmada, Büyük Dil Modelleri (LLM)'nin gelişmesi ile birlikte Karşılaştırmalı Dil-Görüntü Ön Eğitimi (CLIP)'nin önemini belirtmektedir. Bu yöntem, metin ve görseller arasında güçlü bağlantılar kurmak için büyük bir metin ve görüntü verisi ile açıklamaları arasındaki ilişkileri ele almaktadır.

Bu çalışmadaki amacım ve motivasyonum, giderek görselleşen bir dünyada sadece insan kalabalığı sayımını değil, aynı zamanda hayvan, bitki, ağaç ve endüstriyel ürünlerin sayımı gibi çeşitli alanlarda da projenin geniş bir çerçevede kullanılabilmesini sağlamaktır. Bu sayede proje, tek bir alan ile kısıtlı kalmayıp çok yönlü bir şekilde uygulanabilir olmasıdır.

1.1 Çalışmanın önemli parçaları

Çalışmanın öenmli parçaları:Bu çalışma, CLIP (Contrastive Language-Image Pretraining) modelinin kalabalık sayımı konusundaki potansiyelini araştırmaktadır. CLIP modeli, sıfırdan görüntü sınıflandırma ve nesne tespiti gibi tanıma problemlerinde olağanüstü performans sergilemiştir, ancak sayma yeteneği henüz yeterince araştırılmamıştır. Bu çalışmada, sayma işleminin bir regresyon problemi olmasından kaynaklanan zorlukların tanıma problemine dönüştürülmesi üzerine odaklanılmıştır[14].

1.2 önemli noktaları ve ele alınan konular

CLIP Modelinin İncelenmesi

Kalabalık Sayımı İçin Yeni Yaklaşım

Enhanced Blockwise Classification (EBC) Çerçevesi

CLIP-EBC Modeli

Performans Değerlendirmesi

2 LİTERATÜR ARAŞTIRMASI

kalabalık sayımı alanında önemli bir yaklaşım olan CLIP-EBC'yi tanıtan önemli bir çalışma incelenmiştir[14]. Bu çalışma, geleneksel kalabalık sayımı yöntemlerinden farklı olarak CLIP modelini kullanarak yoğunluk haritaları oluşturulması amaçlanmıştır. Bu yaklaşım, kalabalık sayımı alanında yeni bir yaklaşım belirlemekte ve CLIP'in tanıma problemlerindeki başarısını kalabalık sayımı konusunda da kullanabileceğini göstermektedir.

Geliştirilmiş Blok Bazında Sınıflandırma (EBC), bu çalışmanın temelini oluşturmaktadır. EBC, geleneksel sınıflandırma tabanlı kalabalık sayımı yöntemlerinde karşılaşılan zorluklar için tasarlanmıştır. Bu çerçeve, sayımları sağlam karar sınırları öğretmeyi kolaylaştıran tam sayı değerli bloklara dayanmaktadır.

CLIP-EBC'nin önemi, CLIP modelini kullanarak kalabalık sayımı için ayrıntılı dağılım yoğunluk haritaları oluşturabilmesidir. Bu, önceki yöntemlerin başarısız olduğu noktalarda önemli bir ilerleme sağlamaktadır. Özellikle, ShanghaiTech veri setlerinde elde edilen düşük hata oranları, CLIP-EBC'nin etkin kullanımı ve doğruluğunu göstermektedir.

Bu çalışmanın literatüre katkısı, kalabalık sayımı alanında CLIP modelinin potansiyelini ortaya koyması ve geleneksel sınıflandırma tabanlı yöntemlere yenilikçi bir alternatif sunmasıdır. EBC ve CLIP modelinin birleşimi, kalabalık sayımı alanında yeni bir bakış açısı sunarak gelecekteki araştırmalara ilham ve kaynak oluşturabilmesidir.

Proje kapsamında kullanılan üç veri seti vardır: birincisi ShanghaiTech, diğeri Part A ve sonuncusu Part B. ShanghaiTech veri seti, yoğunluklarına göre ikiye ayrılmıştır. Part A, daha yoğun insan topluluklarını içerirken, Part B ise daha az yoğunlukta insan topluluklarını içermektedir[24][18].

ShanghaiTech datasetinden görüntüler

Şekil 1: [18]

Şekil 2: [18]

Şekil 3: [18]

Şekil 4: [18]

İkinci veri seti, Haroon Idrees ve meslektaşları tarafından 2018 yılında tanıtılan UCF-QNRF büyük ölçekli kalabalık sayma veri setidir. Bu veri seti, kalabalık sayma ve yer belirleme yöntemlerini eğitmek ve değerlendirmek için son derece uygundur. Veri seti, 1201 eğitim ve 334 test görüntüsü olmak üzere toplam 1535 görüntüden oluşmaktadır. Derin Konvolüsyonel Sinir Ağları (CNN'ler) için eğitimde etkilidir[11].
UCF-QNRF datasetinden görüntüler:

Şekil 5: [11]

Şekil 6: [11]

Şekil 7: [11]

Şekil 8: [11]

Üçüncü veri seti, Northwestern Politeknik Üniversitesi (NWPU) tarafından oluşturulan NWPU-Crowd veri setidir. İnternetten toplanmış 5,109 görüntü ve 2,133,375 örneklemden oluşan bu büyük ölçekli kalabalık sayma veri seti, yoğun kalabalıkların sayımı için kullanılmaktadır.

NWPU-Crowd Veri Setinden Örnek Görüntüler:

Şekil 9: [11]

Şekil 10: [11]

Şekil 11: [20]

Şekil 12: [20]

3 METODOLOJİ

3.1 ÖNEMLİ KÜTÜPHANELER

3.1.1 Tensorflow

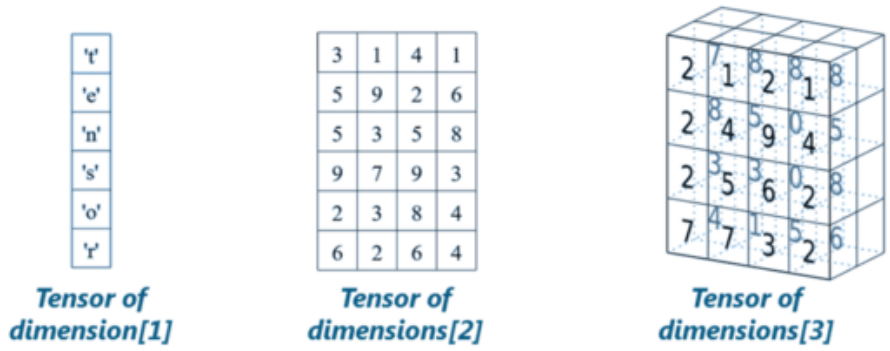
TensorFlow temelde, makine öğrenimi uygulamaları ve sinir ağları geliştirmeye yönelik Python dostu bir açık kaynaklı kütüphanedir.

Google Brain ekibi tarafından oluşturulan ve ilk kez 2015 yılında kamuoyuna sunulan TensorFlow, sayısal hesaplama ve büyük ölçekli makine öğrenimi için açık kaynaklı bir kütüphanedir. TensorFlow, bir dizi makine öğrenimi ve derin öğrenme modeli ve algoritmasını bir araya getirir ve bunları yaygın programlama metaforları aracılığıyla kullanışlı hale getirir. TensorFlow diğer birçok dil için kütüphaneler sağlar, ancak genellikle Python ön plandadır [23].

Tensör nedir ?

Tensorflow'un adı doğrudan çekirdek çerçevesinden türetilmiştir: Tensor. Tensorflow'da tüm hesaplamalar tensörleri içerir. Tensör, tüm veri türlerini temsil eden n boyutlu bir vektör veya matristir. Bir tensördeki tüm değerler, bilinen (veya kısmen bilinen) bir şekle sahip aynı veri tipini içerir. Verinin şekli matrisin veya dizinin boyutluluğudur.

Bir tensör, giriş verilerinden veya bir hesaplamanın sonucundan kaynaklanabilir. TensorFlow'da tüm operaişlemler bir grafik içerisinde gerçekleştirilir[12].

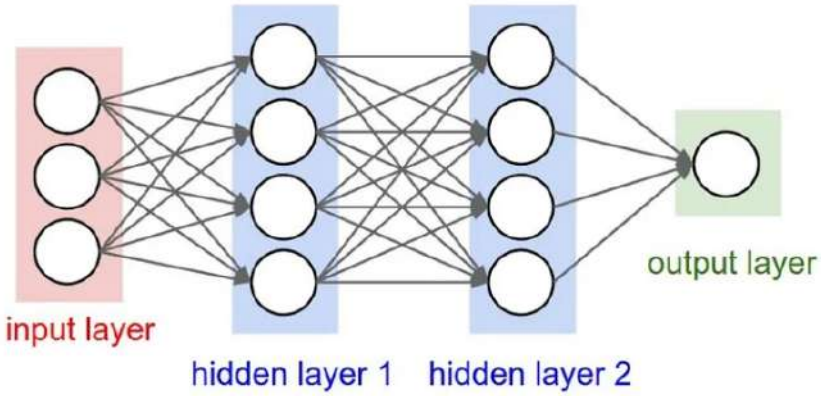


Şekil 13: Tensor Matrisleri [10]

3.1.2 Keras

Keras, Python'da yazılmış ve derin öğrenme için üst düzey bir kütüphane olarak inşa edilmiştir. Theano ve TensorFlow gibi alt yapıları kullanarak, çeşitli derin öğrenme modellerini oluşturmak için temiz ve kullanışlı bir arayüz sunar. Keras, sinir ağlarının geliştirilmesi ve test edilmesi için en yaygın kullanılan üst düzey sinir ağları API'lerinden biri haline gelmiştir.

Keras, kolay ve hızlı yöntemlerle model oluşturulmasına imkân sağlamaktadır. Bu özelliği ile yeni başlayanlar için modellerde, neler değiştirildiğinde nasıl bir etki yaratacağı deneme-yanılma yolu ile öğrenilmektedir. Modelleri merkezi işlem birimi (CPU) ve grafik işlemlerini yürüten işlemcileri (GPU) kullanarak sorunsuz biçimde çalıştırmaktadır. Bu şekilde istenildiği zaman işlemlerin GPU'da yapılarak zaman kazanılmasına destek olmaktadır. Bilgisayarlı görme modelleri olan evrişimsel sinir ağları CNN (Convolutional Neural Network) ve yinelemeli sinir ağlarını RNN (Recurrent Neural Network) desteklemektedir. İçerisinde kütüphane ile ilgili çok fazla kaynağın olması, oluşan ya da oluşabilecek sorunların yanıtına hızlı erişim imkânı sunmaktadır.



Şekil 14: Keras Hidden Layer Örneği [3]

3.1.2 PyTorch

PyTorch, Torch kütüphanesine dayanan açık kaynaklı bir makine öğrenme kütüphanesidir, bilgisayarla görme ve doğal dil işleme gibi uygulamalar için kullanılır. Öncelikle Facebook'un AI Araştırma laboratuvarı tarafından geliştirilmiştir. Grafik işlem birimlerinin gücünü kullanan derin öğrenme modelleri oluştururken kullanılan bir Python kütüphanesidir.

Güçlü GPU hızlandırma desteğiyle tensör hesaplamaları ve teyp tabanlı bir otograd sistemlerinde derin sinir ağları oluşturmaktır. PyTorch'un başarısının arkasındaki temel nedenlerden biri, tamamen Pythonic olması ve sinir ağ modellerini sorunsuz bir şekilde oluşturabilmesidir.

Projelerin içerisinde grafik işlem birimlerini kullanan PyTorch, yapısı gereği sağladığı esneklik ve hız ile de günümüzde oldukça popüler konumdadır[17].

Pythonic in nature:

Pythonic olan bu kütüphane(kodun düzenli ve temiz olması), Python veri bilimi ile kullanılır.

Hesaplamalı grafikler:

PyTorch, dinamik hesaplama grafikleri sunan bir platform sağlar, bu sayede çalışma esnasında değiştirme özelliğine sahiptir.

PyTorch, birçok yaygın derin öğrenme modelinin uygulanmasını kolaylaştıran geniş bir model koleksiyonu olan "TorchVision" ve dil modelleri, görüntü sınıflandırma, nesne tespiti ve diğer çeşitli görevler için önceden eğitilmiş model koleksiyonu olan "TorchHub" gibi ek modüllerle birlikte gelir[9].

Table 1: Keras-Tensorflow-PyTorch karşılaştırma tablosu.

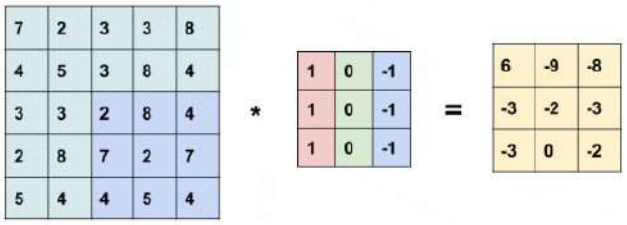
	Keras	TensorFlow	PyTorch
Api Düzeyi	üst düzey API	Hem yüksek hem de düşük seviyeli API'ler	Alt düzey API
Hız	Yavaş	Yüksek	Yüksek
Mimari	Basit, daha okunabilir	Kullanımı pek kolay değil	Karmaşık
Hata Ayıklama	Hata ayıklamaya gerek yok	Hata ayıklamanın zor olması	İyi hata ayıklama yetenekleri
Veri Kümesi Uyumluluğu	Yavaş ve Küçük	Hızlı ve büyük	Yüksek hız ve büyük veri kümeleri
Popülarite Sıralaması	-	-	-
Benzersizlik	Çoklu arka uç desteği	Nesne Algılama İşlevselliği	Esneklik ve Kısa Eğitim Süresi
Oluşturan/Sahibi	Tek başına bir kütüphane değil	Google tarafından oluşturuldu	Facebook tarafından oluşturuldu
Kullanım Kolaylığı	Kullanıcı dostu	Kapsamlı olmayan API	Python dili ile entegre
Kullanılan Hesaplamalı Grafikler	Statik grafikler	Statik grafikler	Dinamik hesaplama grafikleri

3.2 CNN NEDİR ?

CNN genellikle görüntü işlemede kullanılan ve girdi olarak görselleri alan bir derin öğrenme algoritmasıdır. Farklı operasyonlarla görsellerdeki featureları (özellikleri) yakalayan ve onları sınıflandıran bu algoritma farklı katmanlardan oluşmaktadır. Convolutional Layer, Pooling ve Fully Connected olan bu katmanlardan geçen görsel, farklı işlemlere tabii tutularak derin öğrenme modeline girecek kıvama gelir.

Evrişimsel Katman (Convolutional Layer)

Convolutional (evrişim katmanı) CNN algoritmalarında görüntüyü ele alan ilk katmandır. Bilindiği üzere görseller aslında içlerinde belirli değerler taşıyan piksellerden oluşan matrislerdir. Evrişim katmanında da orijinal görsel boyutlarından daha küçük bir filtre görselin üzerinde gezer ve bu görsellerden belirli özellikleri yakalamaya çalışır.



Şekil 15: Evrisimsel temsili resim [6]

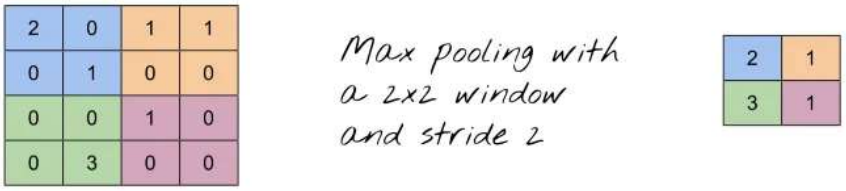
Yukarıda şekil: 15 görüldüğü üzere 3×3'lük bir filtre, 5×5'lik bir görsel üzerinde gezdiriliyor. Çıkan sonuçlar eşitliğin sağ tarafındaki yeni matrisimiz olan feature map üzerine yazılıyor.

Max-Pooling

evrişimli sinir ağlarında (convolutional neural networks - CNNs) kullanılan bir katman türüdür. Evrişimli sinir ağlarında, görüntü işleme gibi görsel veriler üzerinde çalışırken, boyut azaltma ve özellik çıkarma için kullanılır.

Max pooling katmanı, girdi olarak aldığı bir bölgenin (örneğin, bir pencere veya filtre) en büyük değerini çıkararak boyut azaltma işlemi gerçekleştirir. Bu, girdinin belirli bir özellikleri öne çıkararak öznitelik haritasının boyutunu azaltmaya ve işlemin hesaplama yükünü azaltmaya yardımcı olur.

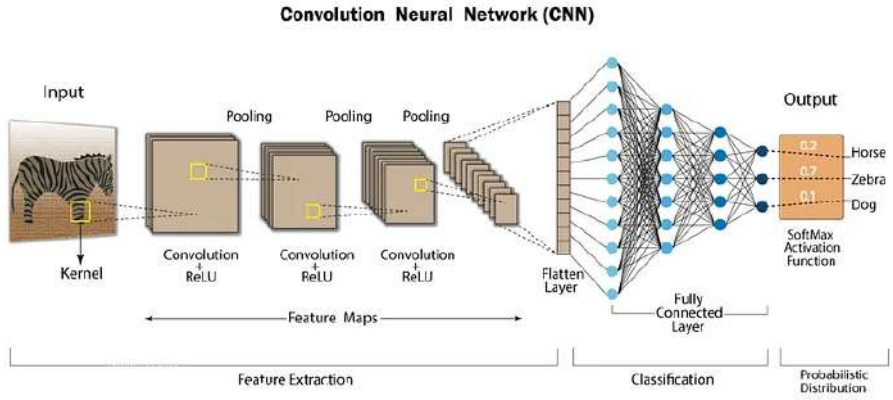
Örneğin, 2x2 boyutunda bir pencereyi (genellikle 2x2 piksel) alıp, bu pencerenin içindeki en büyük değeri alarak çıktı olarak kullanır. Bu işlem, görüntünün boyutunu yarıya indirirken önemli özelliklerin korunmasına yardımcı olur. Bu tür katmanlar genellikle ardışık evrişim katmanları arasında kullanılır ve daha sonraki tam bağlantılı (fully connected) katmanlar için girdi boyutunu azaltır. Bu şekilde, ağın daha hızlı öğrenmesine ve daha fazla genelleme yeteneğine sahip olmasına yardımcı olur.



Şekil 16: maxpooling [6]

Tamamen Bağlantılı Katman ve Evrişimsel Katman

bağlı katman (fully connected layer), genellikle çok katmanlı yapay sinir ağlarında bulunan bir katman türüdür.CNN'lerde, tamamen bağlı katmanlar genellikle ağın sonunda bulunur. Evrişim ve havuzlama katmanlarından sonra, elde edilen özellik haritaları, tamamen bağlı katmanlara bağlanarak sınıflandırma veya regresyon gibi görevler için kullanılır.



Şekil 17: Örnek Convolutional Neural Network — CNN architecture [16]

3.3 CLIP NEDİR ?

CLIP-EBC (Contrastive Language-Image Pretraining with Enhanced Blockwise Classification) yönteminin kalabalık sayımı görevinde nasıl kullanıldığını açıklamaya yöneliktir.

CLIP, OpenAI tarafından geliştirilen ve görüntü-tanımda üstün performans gösteren bir makine öğrenimi modelidir. CLIP, görsellerle ilgili doğal dil açıklamalarını kullanarak önceden eğitilmiş bir modeldir ve sıfırdan öğrenme yeteneğiyle bilinir. Sıfırdan öğrenme, modelin daha önce hiç görmediği kategorileri tanıyabilmesi anlamına gelir.

Sıfır Atışlı Öğrenme Nedir ve Nasıl Çalışır

Modelin öğrenmek üzere eğitildiği kategorilerin etiketlenmiş örneklerinin yokluğunda, sıfır atışlı öğrenme(ZSL) problemleri yardımcı bilgileri kullanır : metinsel açıklamalar, nitelikler, gömülü gösterimler veya eldeki göreve ilişkin diğer anlamsal bilgiler.

Denetimli öğrenmede ve birkaç adımlı öğrenmede (Few-Shot Learning-FSL) model, her sınıfın bir veya daha fazla etiketli örneğini doğrudan gözlemleyerek farklı sınıfları tanımayı öğrenir. Onlara rehberlik edecek bu açık ek açıklamalar olmadan, sıfır adımlı öğrenme, etiketin anlamının daha temel bir şekilde anlaşılmasını gerektirir.

Basit bir benzetme yapmak gerekirse, bir çocuğun bir kuşun neye benzediğini öğrenmek istediğini düşünün. Denetimli öğrenmeye veya FSL'ye benzeyen bir süreçte çocuk, hayvan resimlerinden oluşan bir kitaptaki "kuş" etiketli resimlere bakarak öğrenir. İlerlediğinde, daha önce gördüğü kuş resimlerine benzediği için bir kuşu tanıyacaktır. Ancak ZSL senaryosunda bu tür etiketli örnekler mevcut değildir. Bunun yerine çocuk, kuşlarla ilgili bir ansiklopedi maddesini okuyabilir ve onların havada uçabilen tüyleri, gagaları ve kanatları olan küçük veya orta boy hayvanlar olduğunu öğrenebilir. Daha sonra daha önce hiç görmemiş olmasına rağmen gerçek dünyadaki bir kuşu tanıyabilecektir çünkü kuş kavramını öğrenmiştir.

Transfer öğrenim

Eğitim için gereken zaman ve kaynakların yanı sıra görünmeyen sınıfları tanımlamak için gereken yardımcı bilgi miktarını en aza indirmek amacıyla ZSL, modelleri sıfırdan eğitmek yerine genellikle transfer öğrenmeden (eğitilmiş bir modelin yeni bir görev için yeniden kullanılması) yararlanır.

Transfer öğrenimi, sınıfları ve örnekleri anlamsal yerleştirmeler olarak temsil eden ZSL yöntemlerinde belirgin bir şekilde kullanılır . Örneğin, sıfır atışlı metin sınıflandırması gerçekleştiren bir model, kelimeleri vektör yerleştirmelerine dönüştürmek için zaten çok sayıda dil verisi üzerinde önceden eğitilmiş transformatör tabanlı bir model kullanabilir. Benzer şekilde, sıfır atışlı bir görüntü sınıflandırma modeli, ResNet veya U-Net gibi önceden eğitilmiş bir evrişimli sinir ağını (CNN) yeniden tasarlayabilir , çünkü sınıflandırmayı bilgilendirebilecek önemli görüntü özelliklerini belirlemeye yardımcı olan filtre ağırlıklarını zaten öğrenmiş olacaktır.

Transfer öğrenimi, modelin görülen sınıflara ilişkin bilgisinin, görünmeyen sınıflara ilişkin yardımcı bilgi olarak kullanılabileceği Genelleştirilmiş sıfır vuruşlu öğrenme (GSZL) için özellikle önemlidir. Örneğin, bir nesne algılama modelinin boz ayıları tanımayı zaten öğrendiğini hayal edin. Kutup ayılarının etiketli örneklerini vererek onu kutup ayılarını tanıması için eğitmek yerine, kutup ayılarının beyaz kürklü boz ayılara benzediğini anlayacak şekilde eğitilebilir[4].

Zero-shot learning, bir modelin eğitim sırasında hiç görmediği sınıfları tanımayı veya bu sınıflara ait özellikleri öğrenmeyi amaçlayan bir makine öğrenimi yaklaşımıdır. Bu senaryoda, verilen bilgilere dayanarak Aşağıdaki resimlerden hangi resmin "gavyal" olduğunu tahmin etmeye çalışalım.

Verilen bilgiler doğrultusunda seçim yapınız.
"Yakalanan en büyük gavyal 7 m. uzunluğunda idi, ama normalde uzunlukları 5 metreyi geçmez." "Balık yakalama yönünde evrimleşmiş uzun ve dar ağız yapısı" "Gavyal, tuzlu su timsahından sonra dünyanın en büyük timsah çeşididir."

Şekil 18: [8]

Şekil 19: [8]

Şekil 20: [8]

Şekil 21: [2]

Bu bilgilere dayanarak, en büyük olasılıkla gavyal resminin 4. olduğunu söyleyebilirim. Çünkü 3. bilgi, diğerlerine göre daha spesifik bir özellik veriyor ve gavyalın tuzlu su timsahından sonra dünyanın en büyük timsah çeşididir belirtiyor. Dolayısıyla, bütün resimler arasında seçim yaparken, bu bilgiyi göz önünde bulundurarak 4. resmin gavyal olma olasılığının daha yüksek olduğunu düşünüyorum. Bu nedenle, gavyal resminin 4. olduğunu tahmin ederdim.

3.4 Geliştirilmiş Blok Bazında Sınıflandırma

Geleneksel yöntemler blok bazlı regresyona dayanmaktadır. Ancak, bu Geliştirilmiş Blok Bazında Sınıflandırma (Enhanced Blockwise Classification (EBC)) çerçevesi, her blok içindeki sayı değerini birkaç önceden tanımlanmış aralığa sınıflandırmayı amaçlayan bir fikre dayanmaktadır. Geliştirme, üç ana noktadan gelmektedir: ayrıklaştırma politikası, etiket düzeltme ve kayıp fonksiyonu[11].

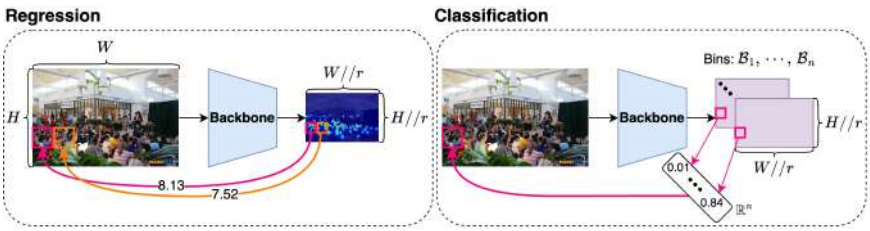
Bu yöntem, görüntüyü küçük bloklara böler ve her bir bloktaki nesne sayısını belirli bir aralığa sınıflandırmayı amaçlar. Bu sınıflandırma, belirli bir blok içindeki nesne sayısını tahmin etmeye odaklanır ve nesnelerin yoğunluğunu ölçmek için kullanılır.

Ayrıklaştırma Politikası: EBC'de kullanılan ayrıklaştırma politikası, nesne sayısını belirli aralıklara sınıflandırmak için belirlenen stratejiyi ifade eder. Bu strateji, nesnelerin dağılımını en iyi şekilde yansıtmak için dikkatlice seçilir.

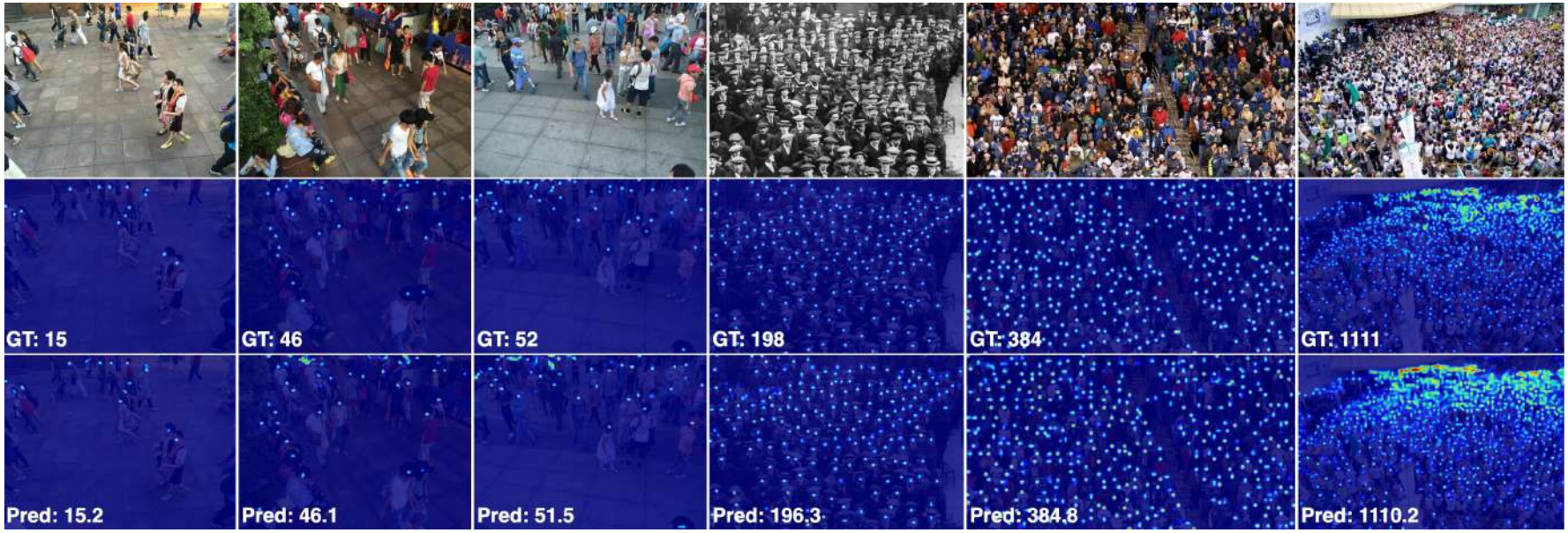
Etiket Düzeltme: EBC, doğru etiketleme için bir düzeltme mekanizması içerir. Bu mekanizma, modelin öğrenirken daha doğru sonuçlar elde etmesine yardımcı olur ve modelin performansını artırır.

Kayıp Fonksiyonu: EBC'nin kayıp fonksiyonu, modelin eğitilirken hata miktarını belirlemek için kullanılır. Bu kayıp fonksiyonu, modelin en iyi sınıflandırma sonuçlarını üretmek için optimize edilir.

Bu üç bileşen, EBC'nin geliştirilmiş performansını sağlar ve nesne sayımı gibi görevlerde daha etkili bir şekilde çalışmasını sağlar.



Şekil 22: EBC [11]



Şekil 23: Proje Çıktısı Örneği [14].

3 C-CNN

Contourlet CNN, görüntü işleme ve analizinde kullanılan bir derin öğrenme mimarisidir. Özellikle doku analizi ve nesne tanıma gibi alanlarda oldukça etkilidir. Contourlet dönüşümü(Contourlet Dönüşümü görüntülerin yumuşak bölgelerinin sınırlarında karşılaşılan kenar yumuşaklıklarını saptamada daha başarılıdır[13].) adı verilen bir matematiksel işlemi kullanarak görüntüyü farklı ölçeklerde ve yönlerde analiz eder ve bu analizleri konvolüsyonel sinir ağları ile birleştirerek karmaşık desenleri ve nesneleri öğrenir.

Contourlet CNN'in Avantajları: Çok Ölçekli Analiz: Görüntüleri farklı ölçeklerde analiz ederek, hem genel doku bilgilerini hem de ince ayrıntıları yakalayabilir. Yönsel Duyarlılık: Görüntüdeki doku ve nesnelerin yönünü de dikkate alarak daha kapsamlı bir analiz yapabilir. Derin Öğrenme Gücü: Konvolüsyonel sinir ağları ile öğrenerek, karmaşık desenleri ve nesneleri yüksek doğrulukla tanıyabilir.

Contourlet CNN'in Kullanım Alanları: Doku Analizi: Tıbbi görüntülerde doku anormalliklerini tespit etme, dokuların türünü ve özelliklerini sınıflandırma. Nesne Tanıma: Fotoğraflarda ve videolarda nesneleri (insanlar, araçlar, hayvanlar vb.) algılama ve sınıflandırma. Görüntü Sınıflandırma: Görüntüleri (doğal manzaralar, şehir manzaraları, iç mekanlar vb.) kategorilere ayırma. Görüntü Geliştirme: Görüntülerdeki gürültüleri azaltma, keskinleştirme, kenar algılama.

maoz ragab'ın projesinden yararlanılmıştır[15].

yavisankar Crowd-Counting-Problem projesinden yaralanılmıştır[22].

4 BULGULAR VE TARTIŞMA

```
[--weight_decay WEIGHT_DECAY]
[--warmup_epochs WARMUP_EPOCHS] [--warmup_lr WARMUP_LR]
[--total_epochs TOTAL_EPOCHS] [--eval_start EVAL_START]
[--eval_freq EVAL_FREQ] [--num_workers NUM_WORKERS]
[--local_rank LOCAL_RANK] [--seed SEED]
trainer.py: error: the following arguments are required: --dataset
An exception has occurred, use %tb to see the full traceback.
SystemExit: 2
```

Kod Bloğu: Proje hataları 1

Projenin Linux ortamında yazılmış olması nedeniyle, Windows ortamında .sh dosyaları çalışmamaktadır. Bu durumda, Windows ortamında projenin çalışması için .sh dosyasından alınan dataset parametresini manuel olarak girilmesi gerekmektedir.

```
def standardize\_dataset\_name {(dataset: str) -> str:}
assert dataset.lower() in available_datasets, f"Dataset {dataset}
is not available."
if dataset.lower() in ["shanghaitech_a", "sha"]:
return "sha"
elif dataset.lower() in ["shanghaitech_b", "shb"]:
return "shb"
elif dataset.lower() in ["ucf_qnrf", "qnrf", "ucf-qnrf"]:
return "qnrf"
elif dataset.lower() in ["nwpu", "nwpu_crowd", "nwpu-crowd"]:
return "nwpu"
else: # dataset.lower() in ["jhu", "jhu_crowd", "jhu_crowd_v2"]
return "jhu"
```

Kod Bloğu: Proje hataları 2

Proje hataları 1, kodunun hata devamı olarak görülebilir. Projenin başlatılması sırasında ilgili parametrelerin otomatik olarak çekilmemesi nedeniyle devam eden bir sorun olarak görülebilir. Bu durumda, ilgili parametrelerin değerlerini manuel olarak kodun ilgili bölümüne girmek gerekmektedir.

```
File ~/Documents/CLIP-EBC-main/utils/train_utils.py:63 in get_loss_fn
assert args.weight_ot is not None and args.weight_tv is not None,
"weight_ot and weight_tv cannot be None when bins is None."
AttributeError: 'Namespace' object has no attribute 'weight_ot'
```

Kod Bloğu: Proje hataları 3

Proje hataları 1, kodunun hata devamı olarak görülebilir. Projenin başlatılması sırasında ilgili parametrelerin girilmemiş olduğu farkedilip Manuel olarak değerler girilmiştir.

`required=True ifadesini  required=False`

`ap.add_argument("-f", required=False)`

ifadesi ise Jupyter Notebook veya diğer not defteri uygulamalarında argparse'ın özel bir durumunu ele alıyor. -f argümanı, Jupyter Notebook'un kendi ihtiyaçları için kullanılır ve eğer belirtilmezse, kod hata vermez.

Bu değişiklikler, betiğinizi veya modülünüzü Jupyter Notebook veya diğer not defteri uygulamalarında daha rahat ve sorunsuz bir şekilde kullanmanızı sağlar. Kullanıcılar istediklerinde argümanları belirtebilirler, belirtmezlerse kod varsayılanları kullanır ve hata almazlar[7].

```
File c:\users\bakid\onedrive\belgeler\clip-ebc-main\datasets\crowd.py:11
from .utils import get_id, generate_density_map
ImportError: attempted relative import with no known parent package \newline
```

Proje sahibi tarafından belirtilen gereksinimlere göre, ilgili kod bölümlerindeki eksiklikler tespit edilmiş ve gerekli düzeltmeler yapılmıştır.

Cuda Sürüm Uyumsuzlukları Ve DDU

CUDA (Compute Unified Device Architecture), GPU (Graphics Processing Unit) için NVIDIA'nın sunduğu C programlama dili üzerinde eklenti olarak kullanıma sunulan bir mimari ve teknolojidir[21].

Ekran Sürücüsü Kaldırıcı(DDU) NEDİR ?

Ekran Sürücüsü Kaldırıcı bilgisayarda yer alan ekran kartı(Nvidia,AMD) sürücüsü veya sürücülerinin kaldırılmasını sağlayan bir program/uygulamadır. indirip kullandığım sürüm Display Driver Uninstaller (DDU) download version 18.0.7.6[19].

```
v2.2.1
Conda
OSX
# conda
conda install pytorch=2.2.1
    torchvision==0.17.1 torchaudio==2.2.1 -c
    pytorch
Linux and Windows
# CUDA 11.8
conda install pytorch==2.2.1
    torchvision==0.17.1 torchaudio==2.2.1
    pytorch-cuda-11.8c pytorch -c nvidia
# CUDA 12.1
conda install pytorch==2.2.1
    torchvision==0.17.1 torchaudio==2.2.1
    pytorch-cuda 12.1 c pytorch -c nvidia
# CPU Only
conda install pytorch=2.2.1
    torchvision==0.17.1 torchaudio==2.2.1
    cpuonly pytorch
Wheel
OSX
pip install torch==2.2.1 torchvision==0.17.1
    torchaudio==2.2.1
Linux and Windows
# ROCM 5.7 (Linux only)
pip install torch==2.2.1 torchvision==0.17.1
    torchaudio=2.2.1 --index-url
    https://download.pytorch.org/whl
# CUDA 11.8
pip install torch==2.2.1 torchvision==0.17.1
    torchaudio==2.2.1 --index-url
    https://download.pytorch.org/whl
# CUDA 12.1
pip install torch==2.2.1 torchvision==0.17.1
    torchaudio=2.2.1 --index-url
    https://download.pytorch.org/whl
# CPU only
pip install torch==2.2.1 torchvision==0.17.1
    torchaudio==2.2.1 --index-url
    https://download.pytorch.org/whl
```

Şekil 24: CUDA-Pytorch [1].

```
packages spyder_kernels\py3compat.py:356 in compat_exec
exec(code, globals, locals)
File c:\users\bakid\onedrive\belgeler\clip-ebc-main7\trainer.py:1
import torch
File ~\anaconda3\Lib\site-packages\torch\__init__.py:21
from torch import _C
ImportError: DLL load failed while importing _C: Belirtilen modül bulunamadı.
```

DLL hatası

Pytorch 2.3.0 sürümü uninstall edilip önerilen sürüm 2.2.1 sürümü tekrardan yüklenilmiş ve DLL sorunu ortadan kaldırılmıştır.

```
get_model
 return vanilla_clip(
 File\OneDrive\Belgeler\CLIP-EBC-main7\models\clip\model.py:165 in_vanilla_clip
 return VanillaCLIP(
 File ~\OneDrive\Belgeler\CLIP-EBC-main7\models\clip\model.py: 57
 in_init
 self.anchor_points = torch.tensor (anchor_points, dtype=torch.float32,
  requires_grad=False).view (1, -1, 1, 1)
 TypeError: must be real number, not NoneType
```

Anchor_points hatası veriyor boş değer dönüyordu devamında aşağıdaki kod parçası denenmiş ve olumlu sonuç vermiştir.

```
self.bins bins
if anchor_points is not None:
self.anchor_points = torch.tensor(anchor_points, dtype=torch.float32,
requires_grad=False).view(1, -1, 1, 1)
print(anchor_points)
else:
print(anchor_points)
return #anchor_points'in None olduğu durumu yönetin
(örneğin,varsayılan bir değer ayarlayın)
```

Daha sonraki denemelerde

```
self.bins bins
self.anchor_points = torch.tensor (anchor_points, dtype=torch.float32,
requires_grad=False).view(1, -1, 1, 1) print(anchor_points)
```

kodunun çalıştığı gözlemlenmiştir sorun çözülmüştür.

```
Hata 2 : File\OneDrive\Belgeler\CLIP-EBC-main7 datasets\crowd.py:67 in __init__
self._check_sanity_()
File ~\OneDrive\Belgeler\CLIP-EBC-main7\datasets\crowd.py:114 in _check_sanity_
assert len(self.image_names) == len(self.label_names) == 300,
f"ShanghaiTech_A train split should have 300 images, but found {len(self.image_names)}."
AssertionError: ShanghaiTech_A train split should have 300 images, but found 0.
```

Self.root(dizin bulamama hatası)

Hata sorunu Label_names ve image_names kısmında okuyamama bulamama sorunları ile alakalıdır bu sorun ise Self.root methodunu ilgilendirmektedir.

```
print(f"Rank {Local_rank} process among (nprocs) processes.")
def run(local_rank: int, nprocs: int, args: ArgumentParser) -> None:
init_seeds(args.seed + local_rank)
setup(local_rank, nprocs)
print(f"Initialized successfully. Training with {nprocs} GPUs.")
device = f"cuda: {Local_rank}" if local_rank != -1 else "cuda:0"
print(f"Using device: {device}.")
ddp = nprocs > 1
if args.truncation is None:
# regression, no truncation. bins, anchor_points = None, None
else:
with open(os.path.join(current_dir, "configs",
f"reduction_{args.reduction}.json"), "r") as f:
config = json.load(f)
[str(args.truncation)] [args.dataset] bins = config["bins"] [args.granularity]

anchor_points = config["anchor_points"] [args.granularity] ["average"]
if args.anchor_points == "c
bins = [(float(b[0]), float(b[1])) for bin bins]
anchor_points = [float(p) for p in anchor_points]
args.bins = bins
args.anchor_points = anchor_points
model = get_model(
backbone="resnet50",
input_size=args.input_size,
reduction=args.reduction,
bins=bins,
anchor_points=anchor_points,
prompt_type=args.prompt_type
)
```

Model Hatası

Orjinalinde `backbone=args.model` olan kod parçası (Orjinal kodda "vgg19_ae" kullanılırken bizim modelimiz "clip_resnet50" olarak seçilmiştir) boş olarak döndüğü için manuel tanımlama yapılmıştır.

```
File\OneDrive\Belgeler\CLIP-EBC-main7\datasets\crowd.py:66 in init
self._make_dataset__()
File\OneDrive\Belgeler\CLIP-EBC-main7\datasets\crowd.py:114 in _make_dataset_
label_names.sort(key=get_id)
File ~\OneDrive\Belgeler\CLIP-EBC-main7\datasets\utils.py:8 in get_id return

int(name_without_extension.split("_")[1])
ValueError: invalid literal for int() with base 10: 'IMG'
```

Hatalı kod:

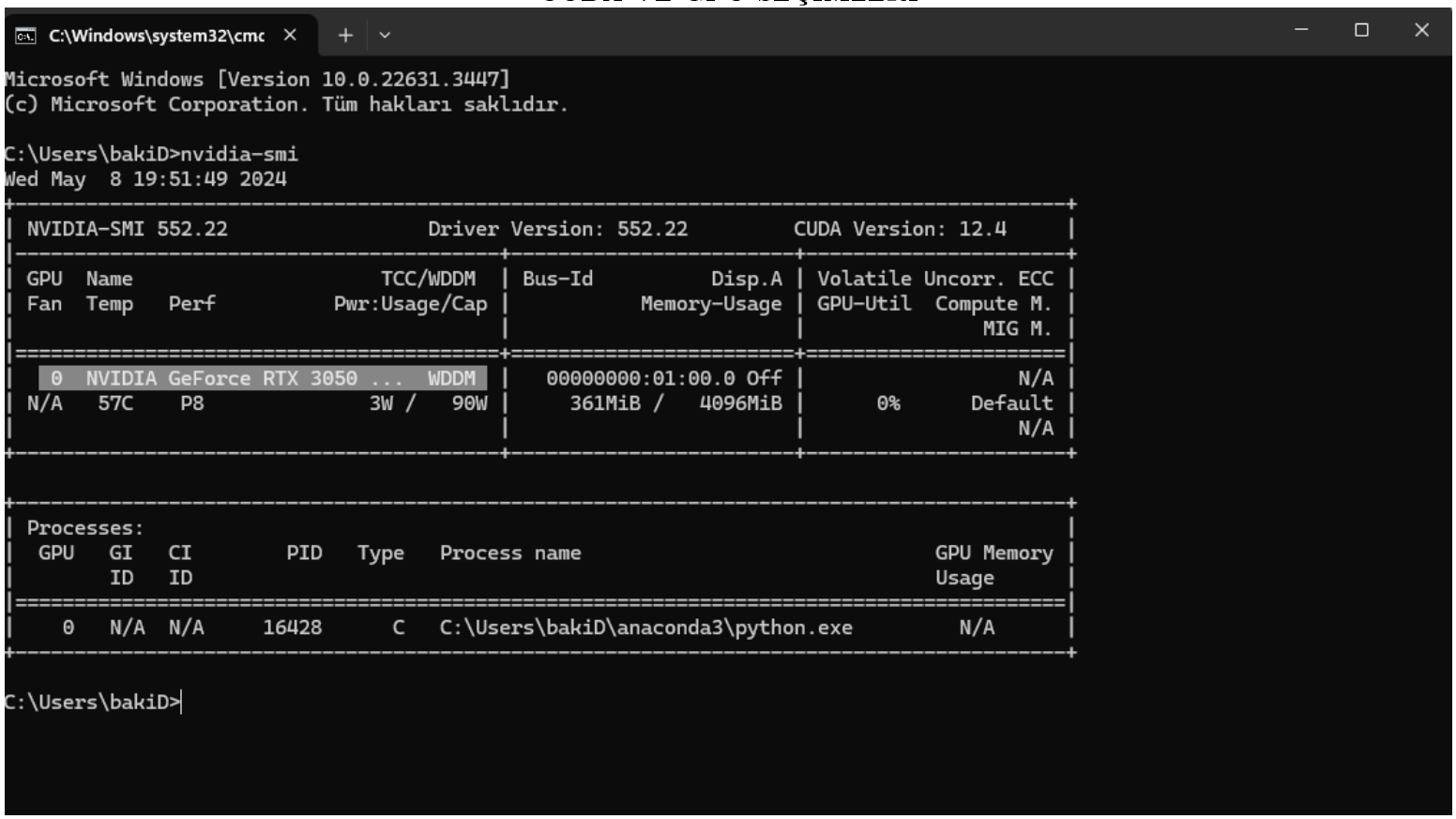
```
def get_id(x: str) -> int:
return int(x.split(".")[0])
```

Düzeltilmiş kod:

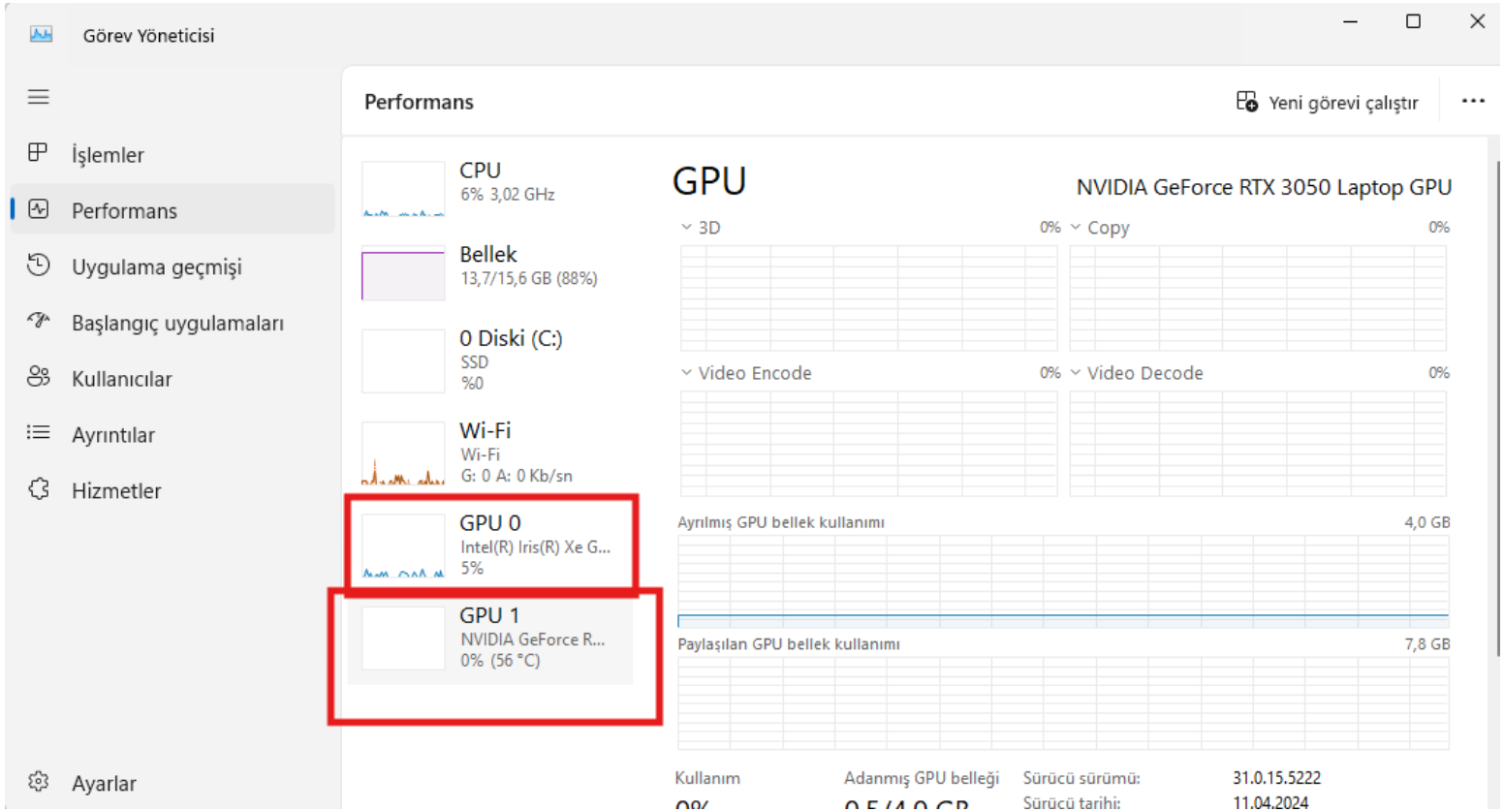
```
def get_id(x: str) -> int:
if x.endswith(".jpg"):
name_without_extension = x.split(".")[0]
return int(name_without_extension.split("_")[1])
else:
name_without_extension = x.split(".")[0]
return int(name_without_extension.split("_")[2])
```

Dosya okuma hatası

Kullanılan veri setindeki resim(.jpg) isimleri "IMG_1" şeklinde iken ground-truth değerleri "GT_IMG_1" şeklindedir, key_id bulma sorunu bu şekilde çözülmüştür.



Şekil 25: nvidia-smi terminal görüntüsü



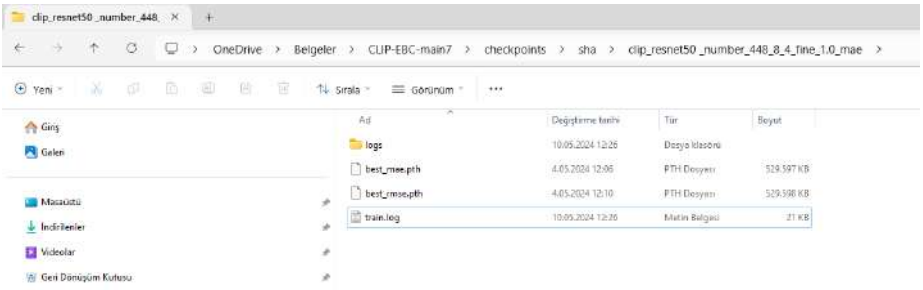
Şekil 26: Windows görev yöneticisi GPU seçenekleri

```
args: ArgumentParser,
model: nn.Module,
optimizer: Adam,
scheduler: LambdaLR,

) -> Tuple[nn.Module, Adam, Union[LambdaLR, None], int, \\
Union[Dict[str, float], None), Dict[str, List[float]]):

ckpt_path = os.path.join(args.ckpt_dir, "best_mae.pth")
print("ne buuuu kiii", args.ckpt_dir)
print("ne buuuu", ckpt_path)
```

Checkpoint Hatası



Şekil 27: Klasor yapısı örneği

5 SONUÇ

Bu çalışma sürecinde, CLIP-EBC yöntemiyle kalabalık sayımı alanında önemli bir ilerleme sağlanmıştır. Ancak, projenin başarılı bir şekilde tamamlanamamasının arkasında çeşitli teknik zorluklar yatmaktadır. Özellikle, grafik kartı ayarlamaları ve eğitim sürecinde karşılaşılan kodda'ki hata problemleri, CLIP modeli gibi karmaşık yapılı modellerin uygulanmasını ve verimli bir şekilde eğitilmesini engellemiştir.

Eğitim sürecinde yaşanan hatalar ve donanım sorunları, projenin ilerlemesini etkilemiş ve istenen sonuçların elde edilmesini engellemiştir. Bu durum, gelecekteki benzer çalışmalar için dikkat edilmesi gereken teknik ve altyapısal zorlukları ortaya çıkarmıştır.

Sonuç olarak, CLIP-EBC yöntemi, kalabalık sayımı alanında yeni bir metodoloji olarak önemli bir potansiyel sunmaktadır. Ancak, projenin tamamlanamamış olması, daha ileri araştırmalar için bu alanda yapılacak çalışmaların daha fazla geliştirilmesi gerektiğini göstermektedir. Bu tür yenilikçi projelerde karşılaşılan her bir zorluk, gelecekteki çalışmalar için önemli bir öğrenme deneyimi olarak değerlendirilmelidir.

Kaynakça

[1] Pytorch. https://pytorch.org/. En son PyTorch sürümü için Python 3.8 veya daha yeni bir sürüm gerekmektedir.

[2] Stok fotoğraf, 2013. Stok Fotoğraf ID: 175949673, Yükleme Tarihi: 7 Ağustos 2013, Kategoriler: Stok Fotoğraflar|Gavyal.

[3] Jaz Allibhai. Building a deep learning model using keras. *Towards Data Science*, 2022.

[4] Dave Bergmann. Zero-shot learning. *IBM Topics*, 2024. https://stackoverflow.com/questions/75503261/an-exception-has-occurred-use-tb-to-see-the-full-traceback-systemexit-2.

[5] bing. yapayzeka resim oluşturucu. https://www.bing.com/images/create/recognizing-objects-and-finding-their-actual-size-/1-664e2694d0f84d9ea6437a3191f47728?darkschemeovr=1&FORM=GUH2CR/.

[6] Bartu Bozkurt. CNN (Convolutional Neural Networks) Nedir? *Medium*, Sep 30 2021.

[7] CBAcnt. An exception has occurred, use systemexit: 2. Sistem Çıkışı: 2. https://stackoverflow.com/questions/75503261/an-exception-has-occurred-use-tb-to-see-the-full-traceback-systemexit-2.

[8] Erdal Ceylan. Dünyanın en İlginç ve en bilinmeyen hayvanları, 2014. Son Güncelleme: 6 Temmuz 2022.

[9] Kaan Cömert. Pytorch nedir? *LinkedIn*, Mayıs 2023.

[10] DevHunter. Tensorflow'un temeli ve mantığı. *DevHunter*, June 2018.

[11] Haroon Idrees, Muhmmad Tayyab, Kishan Athrey, Dong Zhang, Somaya Al-Maadeed, Nasir Rajpoot, and Mubarak Shah. Composition loss for counting, density map estimation and localization in dense crowds. In *Proceedings of the European conference on computer vision (ECCV)*, pages 532–546, 2018.

[12] Daniel Johnson. Tensorflow temelleri: Tensör, Şekil, tür, oturumlar ve operaları. *Guru99*, December 2023.

[13] Bedrettin Erbil Konuk. Hyperspectral data classification using contourlet transform. Master's thesis, Bilişim Enstitüsü, 2016.

[14] Yiming Ma, Victor Sanchez, and Tanaya Guha. Clip-ebc: Clip can count accurately through efficient backpropagation and calibrated loss. *Papers With Code*, 2024. https://paperswithcode.com/paper/clip-ebc-clip-can-count-accurately-through.

[15] Maozed Ragab. Crowd counting part a: val_mae: 10.9 - val_mse: 384. https://www.kaggle.com/code/maozragab/crowd-counting-part-a-val-mae-10.9-val-mse-384, 2021.

[16] Nafiz Shahriar. What is convolutional neural network — cnn (deep learning). *Medium*, Feb 1 2023.

[17] TalentGrid. Pytorch nedir? https://talentgrid.io/tr/pytorch-nedir/, Ocak 2024.

[18] Thai Thien. Shanghaitech. https://www.kaggle.com/datasets/tthien/shanghaitech/data, 2016. Dataset appeared in CVPR 2016 paper MCNN.

[19] wagnard. Ddu. https://www.guru3d.com/download/display-driver-uninstaller-download/. Display Driver Uninstaller (DDU) download version 18.0.7.6.

[20] Qi Wang, Junyu Gao, Wei Lin, and Xuelong Li. Nwpu-crowd: A large-scale benchmark for crowd counting and localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[21] Wikipedia contributors. Cuda — Wikipedia, the free encyclopedia. https://tr.wikipedia.org/wiki/CUDA, 2022.

[22] yavisankar. Crowd-counting-problem. https://github.com/yavisankar/Crowd-Counting-Problem/tree/main, 2021.

[23] Serdar Yegulalp. What is tensorflow? the machine learning library explained. *InfoWorld*, January 2024. https://www.infoworld.com/article/3278008/what-is-tensorflow-the-machine-learning-library-explained.html.

[24] Yingying Zhang, Desen Zhou, Siqin Chen, Shenghua Gao, and Yi Ma. Single-image crowd counting via multi-column convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 589–597, 2016.