T.C.
KONYA TEKNİK ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

YÜKSEK ÇÖZÜNÜRLÜKLÜ UYDU GÖRÜNTÜ VERİLERİ KULLANILARAK BİNA VE YOL SINIFLARININ DERİN ÖĞRENME YÖNTEMİYLE BELİRLENMESİ

Duygu ARIKAN

DOKTORA TEZİ

Harita Mühendisliği Anabilim Dalı

Temmuz-2024
KONYA

TEZ KABUL VE ONAYI

Duygu ARIKAN tarafından hazırlanan "Yüksek Çözünürlüklü Uydu Görüntü Verileri Kullanılarak Bina ve Yol Sınıflarının Derin Öğrenme Yöntemiyle Belirlenmesi" adlı tez çalışması 27/06/2024 tarihinde aşağıdaki jüri tarafından oy birliği ile Konya Teknik Üniversitesi Lisansüstü Eğitim Enstitüsü Harita Mühendisliği Anabilim Dalı'nda DOKTORA TEZİ olarak kabul edilmiştir.

Jüri Üyeleri	**İmza**
Başkan Prof. Dr. Ferruh YILDIZ Konya Teknik Üniversitesi	……..
Üye Prof. Şinasi Kaya İstanbul Teknik Üniversitesi	……..
Üye Prof. Semih EKERCİN Antalya Bilim Üniversitesi	……..
Üye Prof. Dr. Hakan KARABÖRK Konya Teknik Üniversitesi	……..
Üye Doç. Dr. Ömer Kaan BAYKAN Konya Teknik Üniversitesi	……..

Yukarıdaki sonucu onaylarım.

Prof. Dr. Mevlüt UYAN
Enstitü Müdürü

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

İmza

Duygu ARIKAN

Tarih: 27.06.2024

DOKTORA TEZİ

YÜKSEK ÇÖZÜNÜRLÜKLÜ UYDU GÖRÜNTÜ VERİLERİ KULLANILARAK BİNA VE YOL SINIFLARININ DERİN ÖĞRENME YÖNTEMİYLE BELİRLENMESİ

Duygu ARIKAN

Konya Teknik Üniversitesi
Lisansüstü Eğitim Enstitüsü
Harita Mühendisliği Anabilim Dalı

Danışman: Prof. Dr. Ferruh YILDIZ

Yıl, 160 Sayfa

Jüri
Prof. Dr. Ferruh YILDIZ
Prof. Dr. Hakan KARABÖRK
Doç. Dr. Ömer Kaan BAYKAN
Prof. Dr. Semih EKERCİN
Prof. Dr. Şinasi KAYA

Yeryüzü üzerine tesis edilmiş binalar ya da yol ağları birçok uygulamaya konu ya da altlık olan coğrafi nesnelerdir. Bu uygulamaların başarısı ise, binaların ve yol ağlarının güncel olmasına bağlı bir durumdur. Araştırmacılar yapılan çalışmalarda veri kaynağı olarak çoğunlukla uydu veya hava fotoğraflarını kullanmakta olup bu coğrafi nesnelerin otomatik olarak belirlenmesi konusuna odaklanmışlardır. Hava fotoğrafı veya uydu görüntüleri üzerinden sınıflandırma işlemi için çeşitli yöntemler geliştirilmişlerdir. Son yüzyılın en önemli araştırma konularını oluşturan yapay zekâ, makine öğrenmesi ve derin öğrenme yöntemleri de bu gelişime dahil olmuştur. Yakın zamanlarda popüler olan ve yapay zekanın alt dalı olan derin öğrenme teknikleri, veriler arasındaki karmaşık ilişkileri daha iyi anlamak için kullanılmakta ve verilerin içerdiği bilinmeyen özellikleri daha doğru bir şekilde belirleyebilmektedir.

Bu tez çalışmasının amacı derin öğrenme tekniği içerisinde yer alan evrişimli sinir ağlarını kullanarak, bina ve yol ağlarının otomatik olarak çıkarılmasını sağlamaktır. Eğitim verisi olarak Inria hava görüntüsü kullanılmış ve model eğitilmiştir. Test verisi olarak Türkiye'ye ait Göktürk-1 uydu görüntüleri kullanılmıştır. Evrişimsel sinir ağı modelinde U-Net mimarisi seçilmiştir. Bu mimari içerisinde çeşitli optimizasyonlar, batch boyutları ve öğrenme oranlarının tahmin görüntülerine etkisi incelenmiştir. SGD (Stokastik Gradyan İnişi), Adam (Uyarlanabilir Moment Tahmini), RMSprop (Karekök Ortalama Kare Yayılması), Nadam (Nesterov Hızlandırılmış Uyarlanabilir Moment Tahmini), Adagrad (Uyarlanabilir Gradyan Algoritması) ve Adadelta (Uyarlanabilir Delta) optimizasyonları kullanılmıştır. Batch boyutu ve öğrenme oranlarındaki değişimi izlemek amacıyla, Adam optimizasyonu sabit olacak şekilde diğer parametreler gözlemlenmiştir. Bu amaçla batch boyutu için 3 farklı durum boyutu (8,16,32) ele alınmıştır. Öğrenme oranında ise 1e-4 (10^{-4}) ve 1e-3 (10^{-3}) durumları incelenmiştir. Ayrıca, eğitim sayısının miktarındaki değişimlerin de sonuca etkileri değerlendirilmiştir. Bu nedenle eğitim sayısı 1000,2500 ve 5000 görüntü olacak şekilde 3 gruptan oluşmaktadır. Tüm değişimleri 10 epok, 25 epok, 50 epok ve 100 epoktaki sonuçları şeklinde incelenmiştir. Doğruluk ölçütleri olarak hassasiyet, duyarlık, F1 skoru, doğruluk dice ve jaccard metrikleri kullanılmıştır.

Elde edilen bulgular doğrultusunda, batch boyutu 16 ve 1e-3 öğrenme oranının, model performansını optimize etmede etkili olduğu gözlemlenmiştir. Bu bilgi elde edildikten sonra eğitim sayısı 1000 için 6 farklı optimizasyon sonuçları değerlendirilmiştir. Bu optimizasyonlarda en iyi sonuç Adam, daha sonra ise RMSprop optimizasyonuna aittir. 100 epok sonucunda Adam optimizasyonunun doğruluğu

%82.23, RMSprop optimizasyonun %78.76 iken diğer 4 optimizasyonun %40-50 civarında olduğu tespit edilmiştir. Adam optimizasyonunun Jaccard ve Dice metrikleri sırasıyla %81.89 ve %91.56'dir. RMSprop'ta ise bu metrikler sırasıyla %70.84 ve %85.04'dir. Diğer optimizasyonların Jaccard ve Dice doğruluk ölçütleri ortalaması yaklaşık %30-40 seviyelerindedir. Eğitim sayısı 1000 iken tahmin edilen görüntülerin içerinde bazı yapılar tesis edilemediği belirlenmiştir. Eğitim sayısının arttırılmasıyla bu durumun ortadan kalktığı anlaşılmıştır. Çünkü eğitim sayısı 5000 görüntüden oluşan ve Adam ile optimize edilen görüntülerin tahmin sonuçları gayet başarılıdır. Bu başarılı sonucu 2018,2019,2020 ve 2021 yıllarında farklı zaman dilimlerinde (yaz mevsimi-kış mevsimi) alınan görüntülere uygulanmıştır. Ve kış mevsiminde alınan görüntülerin tahmin sonuçlarının kötü olduğu tespit edilmiştir. U-Net mimarisi ile binaların tespit edilmesinin mümkün olduğu açıkça anlaşılmaktadır. Yol ağlarında ise, bina gibi başarılı sonuçlar üretememiştir. Bunun temel nedeni ise, görüntü üzerinde yol ağlarına yakın çok fazla bilgi içermesindendir. Çünkü yol ağlarının, yakınında araçların olması ya da ağaç gibi bitki örtüsünün yolu kapatmasından ötürü sonuçlara negatif olarak yansımıştır.

Anahtar Kelimeler: Bina çıkarımı, derin öğrenme, evrişimli sinir ağları, yol ağları çıkarımı, segmentasyon, U-Net mimarisi

ABSTRACT

PhD THESIS

DETERMİNATİON OF BUİLDİNG AND ROAD CLASSES USİNG HİGH-RESOLUTİON SATELLİTE IMAGE DATA WİTH A DEEP LEARNİNG METHOD

Duygu ARIKAN

Konya Technical University
Institute of Graduate Studies
Department of Geomatics Engineering

Advisor: Prof. Dr. Ferruh YILDIZ

Year, 160 Pages

Jury
Prof. Dr. Ferruh YILDIZ
Prof. Dr. Hakan KARABÖRK
Assoc. Prof. Dr. Ömer Kaan BAYKAN
Prof. Dr. Semih EKERCİN
Prof. Dr. Şinasi KAYA

Buildings or road networks established on the earth's surface are geographic objects that are the subject or basis of many applications. The success of these applications depends on the currency of buildings and road networks. In their studies, researchers frequently use satellite or aerial photographs as data sources and focus on the automatic detection of these geographic objects. Various methods have been developed for the classification process from aerial photographs or satellite images. Artificial intelligence, machine learning, and deep learning methods, which have been among the most important research topics of the last century, have also been included in this development. Recently popular deep learning techniques, a subset of artificial intelligence, are used to better understand the complex relationships between data and to more accurately identify the unknown features contained within the data.

The aim of this thesis study is to extract buildings and road networks automatically using convolutional neural networks, a technique within deep learning. Inria aerial images were used as training data, and the model was trained. Göktürk-1 satellite images belonging to Turkey were used as test data. The U-Net architecture was selected for the convolutional neural network model. Within this architecture, various optimizations, batch sizes, and learning rates were investigated for their effects on predicted images. SGD (Stochastic Gradient Descent), Adam (Adaptive Moment Estimation), RMSprop (Root Mean Square Propagation), Nadam (Nesterov-Accelerated Adaptive Moment Estimation), Adagrad (Adaptive Gradient Algorithm) ve Adadelta (Adaptive Delta) optimizations were used. To monitor changes in batch size and learning rate, other parameters were observed while keeping the Adam optimization constant. For this purpose, three different batch sizes (8, 16, 32) were considered. Learning rates of 1e-4 and 1e-3 were investigated. Additionally, changes in the number of training iterations were evaluated. Therefore, the number of training iterations consists of three groups: 1000, 2500, and 5000 images. All changes were examined in the results at 10 epochs, 25 epochs, 50 epochs, and 100 epochs. Precision, recall, F1 score, accuracy, Dice, and Jaccard metrics were used as accuracy criteria.

Based on the obtained findings, a batch size of 16 and a learning rate of 1e-3 have been observed to be effective in optimizing model performance. After obtaining this information, the results of 6 different optimizations were evaluated for 1000 training iterations. In these optimizations, the best result belonged to Adam, followed by RMSprop optimization. At 100 epochs, the accuracy of Adam optimization was

82.23%, RMSprop optimization was 78.75%, while the other 4 optimizations were around 40-50%. The Jaccard and Dice metrics for Adam optimization were 81.89% and 88.25%, respectively. For RMSprop, these metrics were 70.84% and 85.08%, respectively. The average of Jaccard and Dice accuracy criteria for other optimizations is approximately 30-40%. It was observed that some structures could not be detected in the predicted images when the number of training iterations was 1000. This issue was resolved with an increase in the number of training iterations. Because the images consisting of 5000 training iterations and optimized with Adam yielded quite successful prediction results. This successful result was applied to images taken at different times (summer-winter) in 2018, 2019, 2020, and 2021. It was observed that the prediction results of images taken in winter were poor. It is clearly understood that buildings can be detected with the U-Net architecture. However, successful results have not been achieved for road networks. The main reason for this is that there is too much information near road networks in the image. This is because the presence of vehicles near road networks or vegetation such as trees blocking the road negatively affected the results.

Keywords: Building extraction, deep learning, convolutional neural networks, road networks extraction, segmentation, U-Net architecture

Aileme..

ÖNSÖZ

Tez konumun belirlenmesinde, bilgisini ve yorumlarını eksik etmeyen, çeşitli kaynaklara ulaşmamda yardımını esirgemeyen ve tüm işlemlerde her türlü desteğini çekinmeden bana sunan, öğrencisi olduğum için her zaman gurur duyacağım değerli danışmanım Sayın Prof. Dr. Ferruh YILDIZ'a çok teşekkür ederim. Tecrübesinden, bilgisinden her zaman destek aldığım Prof. Dr. Hakan KARABÖRK'e ve Doç. Dr. Ömer Kaan BAYKAN'a, adını burada yazamadığım tüm hocalarıma, arkadaşlarıma teşekkürü bir borç bilirim.

Bugünlere gelmem de maddi ve manevi en büyük pay sahibi olan ve beni her konuda cesaretlendiren, sonsuz desteklerini esirgemeyen değerli ailemin tüm fertlerine ve her daim yanımda olan hayat arkadaşıma gönülden teşekkür ederim.

Duygu ARIKAN
KONYA-2024

İÇİNDEKİLER

SİMGELER VE KISALTMALAR

Simgeler

θ_t: t zamanındaki güncellenmek istenen parametrenin değeri

α: öğrenme oranı

$\nabla J(Q_t)$: Kayıp fonksiyonunun θ_t noktasındaki gradyanı, yani parametrelerin ne kadar değiştirilmesi gerektiğini gösteren vektör,

$\theta_{t,i}$: t zamanda güncellenmek istenen i parametrenin değeri,

$G_{t,ii}$: t zamanda, i parametre bileşeninin geçmiş gradyan karelerinin toplamı,

ϵ: sıfıra bölme hatasını önlemek için seçilen küçük bir değer

$\nabla J(\theta_{t,i})$: Kayıp fonksiyonunun i parametreye göre türevi,

$\widehat{m_t}$: t zamanındaki hareketli ortalama gradyan

$\widehat{v_t}$: t zamanındaki hareketli ortalama gradyanın karesi

$\widehat{g_t}$: t zamanındaki ikinci moment gradyan

β: Nesterov momentumun ağırlıklandırma faktörü

Kısaltmalar

Adadelta : Adaptive Delta (Uyarlanabilir Delta),

Adagrad : Adaptive Gradient Algorithm (Uyarlanabilir Gradyan Algoritması),

Adam : Adaptive Moment Estimation (Uyarlanabilir Moment Tahmini),

ASPP : Atrous Spatial Pyramid Pooling (Genişletilmiş Uzaysal Piramit Havuzu),

BS : Batch Size (Küme boyutu),

cm : Santimetre

CNN : Convolutional Neural Network (Evrişimli Sinir Ağı - ESA),

DBN : Deep Belief Network (Derin İnanç Ağları – DİA),

DEM : Digital Elevation Model (Sayısal Yükseklik Modeli),

FCN : Fully Convolutional Network (Tam Evrişimli Ağ),

FN : False Negative (Yanlış Negatif),

FP : False Positive (Yanlış Pozitif),

HOG : Histogram of Oriented Gradients (Yönlendirilmiş Gradyanlar Histogramı),

HGM : Harita Genel Müdürlüğü

IoU : Intersection over union /Jaccard skoru (Kesişim Bölü Birleşim),

LIDAR: Light Detection and Ranging (Işık Algılama ve Ölçme),

LSTM : Long Short-Term Memory (Uzun Kısa Süreli Bellek),

m : Metre

MBR-Net : Minimum Bounding Rectangle Network (Minimum Sınırlayıcı Dikdörtgen Ağı),

NAIP : National Agriculture Imagery Program (Ulusal Tarım Görüntüleme Programı),

Nadam : Nesterov-Accelerated Adaptive Moment Estimation (Nesterov Hızlandırılmış Uyarlanabilir Moment Tahmini),

NIR : Near-Infrared (Yakın Kızılötesi)

RAM: Random Access Memory (Rastgele Erişim Belleği),

PAN : Pixel Aggregation Network (Piksel Toplama Ağı),

RBM : Restricted Boltzmann Machines (Kısıtlı/Sınırlı Boltzmann Makineleri),

RCFN : Random Convolutional Features Network (Rastgele Konvolüsyon Özellikleri Ağı),

ReLU : Rectified Linear Unit (Doğrultulmuş Doğrusal Birim),

RGB : Red, Green, Blue (Kırmızı, Yeşil, Mavi),

Rmsprop : Root Mean Square Propagation(Karekök Ortalama Kare Yayılması),

RNN : Recurrent Neural Network (Tekrarlayan Sinir Ağı), SVM: Support Vector Machine (Destek Vektör Makinesi – DVM)

Seg-Net : Segmentation Network (Segmentasyon Ağı),

SGD : Stochastic Gradient Descent (Stokastik Gradyan İnişi),

TN : True Negative (Doğru Negatif),

TP : True Positive (Doğru Pozitif),

VHR : Very High Resolution (Çok Yüksek Çözünürlük),

μm : Mikrometre

1. GİRİŞ

Dünya şartlarının evrimi ve nüfusun hızla artışı, insanların yerleşim bölgelerini düzensiz bir biçimde oluşturmasında hız kazandırmıştır. Kentleşmedeki yaşanan bu değişim küresel çapta hızla gelişen ancak sıklıkla planlama eksikliği yaşanan şehirleşme problemlerine yol açmıştır. Kentleşmenin sürekli genişlemesi nedeniyle plansız bir şekilde büyüdüğü görülmektedir. Kentsel coğrafi veri tabanlarının en önemli ve en sık güncellenen parçaları binalar ve yollardır (Yong Li ve Wu, 2008). Günümüzde mevcut durumun değerlendirilmesi, yapılaşmanın takibi ile karar vericilerin kentlerin gelişim yönünü tayin etmelerinin önemi her geçen gün artmaktadır (Bozkurt, 2018; Üzümlü, 2019). Binaların, yolların tespit edilmesi ya da sınıflandırılması mimarlara, şehir plancılarına ve karar vericiler için kentsel alanların geliştirilmesi, tasarımı ve sürdürülebilirliği konusunda bilinçli kararlar almalarına katkı sağlayabilir (El-Mekawy ve ark., 2012; Shaohua Wang ve ark., 2024).

Binalarda olduğu gibi benzer şekilde yol ağları da şehir planlaması ve altyapı geliştirmesi kapsamında önemli bir rol oynamaktadır. Çünkü etkili bir ulaşım ağı, toplulukların gelişimini destekler ve yaşam kalitesini artırabilir.

Kentlerin değişen ihtiyaçlarına daha etkili bir şekilde yanıt verebilmek için, bölgenin güncel haritalarının oluşturulması ve güncellenmesi oldukça önemlidir. Ancak geleneksel yöntemlerle üretilen haritalar, genellikle maliyetli, zaman alıcı ve pratik olmaktan uzaktır. Bu durum, kentsel planlama ve yönetim süreçlerinde eksikliklere yol açması söz konusudur. Uzaktan algılama teknolojilerinin kullanımı, bu açığı kapatmak için önemli bir çözüm sunmaktadır. Uzaktan algılama, hızlı ve maliyet etkin bir şekilde geniş alanları tarayarak detaylı ve güncel haritaların elde edilmesini sağlar. Bu da planlama süreçlerinin daha verimli ve etkili bir şekilde yürütülmesine olanak tanır (Üzümlü, 2019).Uzaktan algılama teknolojisinin uygulanması, kentsel planlama ve inşaat, harita güncelleme, çevre değişikliği izleme, akıllı şehir ve diğer birçok alana uygulanabilecek binalara ve diğer yapay özelliklere ilişkin bilgileri hızlı ve doğru bir şekilde elde edebilir (Yong Li ve Wu, 2008; Shaohua Wang ve ark., 2024; Zheng ve ark., 2020). Elde edilen bina verileriyle, nüfus tahmini, kentsel izleme, değişiklik tespiti ve sosyo-ekonomik çalışmalarda diğer birçok uygulama alanlarında kullanılmaktadır (Ok, 2013; Shaohua Wang ve ark., 2024; H. L. Yang ve ark., 2018). Yol haritalarıyla şehir planlama, trafik yönetimi, afet yönetimi, askeri müdahaleler gibi uygulamalarda amaca ulaşım kolay hale gelmektedir (Öztürk, 2023).

Uydu teknolojilerindeki gelişmelerle birlikte verilere erişim imkanı ve verilerin uygun maliyetli olması, uzaktan algılama görüntülerinden yararlanarak bilgi çıkarmak avantaj olarak görülmektedir (H. L. Yang ve ark., 2018). Özellikle çok yüksek çözünürlükte multispektral görüntü üreten uydular, büyük ölçekli ve çok miktarda detay içeren haritalar elde edilmesi sağlamaktadır (Üzümlü, 2019). Bu haritalar; doğal kaynaklar, coğrafya, yer bilimleri, savunma yönetimi ve izleme gibi çeşitli alanlarda altlık harita olarak kullanılmaktadır (Saralioglu ve Gungor, 2022). Farklı birçok alanın ihtiyaçlarını karşılamak için bu verilen daima güncel olması gerekmektedir. İşlemlerin manuel olarak yapılması, toplam alan düşünüldüğünde hem çok büyük bir zaman kaybına, sürekli insan çabasına ve çok ağır bir maddi yük anlamına gelmektedir. Bu yüzden birçok araştırmacı, minimal veya hiç insan müdahalesi gerektirmeyen yöntemler geliştirmek için çaba sarf etmiştir. Hava fotoğrafı ya da uydu verilerinden nesneleri tespit etmek için geliştirilmiş bir Canny kenar algılama operatörü (Ali ve Clausi, 2001; Wei ve ark., 2004) ve makine öğrenmesi yöntemlerinden çeşitli algoritmalara (Belgiu ve Drăguţ, 2016; Hänsch ve Hellwich, 2010) dayalı olarak bilgilerinin çıkarılması için insan-bilgisayar etkileşimli yarı otomatik yöntemler önerilmiştir. Ancak bu yöntemlerde büyük bir işçilik durumu söz konusudur. Ayrıca kenar tespit operatörü ve makine öğrenme tekniklerine dayanan nesne çıkarma doğruluğu, uygulama gereksinimlerini tam olarak karşılayamamaktadır (Shaohua Wang ve ark., 2024).

Bu işlemlerin hızlı bir şekilde gerçekleştirilmesi ve anlamlı sonuçlar üretmek için çeşitli derin öğrenme yöntemlerinden yararlanılmaktadır. Son zamanlarda giderek popülerleşen derin öğrenme yöntemleri, uzaktan algılama problemlerinin çözümüne önemli katkılar sağlamaktadır (Shuyang Wang ve ark., 2020). Derin öğrenme yöntemleri, bilgisayarlı görüş alanında genellikle görüntü sınıflandırma (Krizhevsky ve ark., 2017; Kussul ve ark., 2017), nesne tespiti, hedef izleme (F. Gao ve ark., 2019; Yuan ve ark., 2020) ve görüntü segmentasyonu (Chen ve ark., 2017; Long ve ark., 2015) gibi görevlerde kullanılmaktadır. Bu alanda giderek artan sayıda çeşitli bilimsel çalışmalar bulunmaktadır (Kattenborn ve ark., 2019; Kussul ve ark., 2017; Mohanty ve ark., 2016). Ghosh ve ark. (2018), uzaktan algılama görüntülerinde arazi örtüsü otomatik sınıflandırması için genişletilmiş konvolüsyon katmanlı bir U-Net ağ mimarisi önerdi. Seferbekov ve ark. ise benzer amaç doğrultusunda özellik piramit tabanlı tam evrişimli ağ (FCN) önerdi. Ağ içerisinde ImageNet, ResNet50 kodlayıcısı üzerinde önceden eğitilmiş kod çözücü yapısı yer almaktadır.

Segmentasyon da kullanılan çeşitli mimarilerin farklı amaçları vardır. U-Net, yüzey ve derin özelliklerin birleşimiyle segmentasyon doğruluğunu artırırken, DeepLab daha geniş bir alanı kapsamak için genişlemiş evrişim yöntemini kullanır (Chen ve ark., 2017; Chen ve ark., 2018; Ronneberger ve ark., 2015). Piksel toplama ağı (PAN), özellik piramidini dikkat mekanizması ve uzamsal piramitle birleştirerek farklı ölçeklerdeki özellikleri entegre eder (W. Wang ve ark., 2019). UNet++, temelini U-Net'ten alan atlama bağlantılarının yapısını yeniden düzenleyerek, kodlayıcı ve kod çözücü arasında daha esnek bir özellik birleştirme sağlar, böylece ağın performansını artırır (Zhou ve ark., 2019).

Derin öğrenmede kullanılan veri kaynağına bağlı olarak, bina ve yol çıkarma tekniklerini üç gruba ayırmak mümkündür. Birincisi hava veya uydu görüntüsü verileri, İkincisi üç boyutlu bilgileri kullanan veri kaynakları ve sonuncusu her iki veri türünün birleştirilmesidir(Shaohua Wang ve ark., 2024). Bununla birlikte, son zamanki çalışmalarda LIDAR görüntü verilerinden gelen bilgilerin kullanılması gündemdedir (Maltezos ve ark., 2017).

Bu tez çalışması beş bölümden oluşmaktadır. Birinci bölümde konuya ilişkin olarak genel bilgiler verilmiş olup, ikinci bölümde literatür taraması yapılmıştır. Üçüncü bölümde uygulamada kullanılan veriler ve yöntemler açıklanmıştır. Dördüncü bölümde uygulamalar sonucu elde edilen bulgular irdelenmiştir. Beşinci bölümde ise bu çalışma neticesinde varılan sonuçlar verilmiştir.

1.1. Tezin Amacı

Bu tez çalışmasının temel amacı, yol ve bina çıkarım işlemlerini hızlı ve kolay bir şekilde gerçekleştirmektir. Kısa sürede ve kolay bir şekilde arazi hakkında bilgi toplamak ve sınıflandırma işlemini gerçekleştirmek her açıdan önem arz etmektedir.

Literatürde, görüntülerin özelliklerinden yararlanarak bina ve yol çıkarım işlemlerinin gerçekleştirildiği görülmektedir. Bu amaçla, çalışma verileri eğitim ve test setlerine bölünmektedir. Ancak, bu görüntülerin sınırlı coğrafi alanları kapsaması nedeniyle, mevcut değerlendirme prosedürleri, yöntemlerin farklı bağlamlara veya daha soyut anlamsal sınıflara nasıl genelleştirilebileceğini kapsamlı bir şekilde değerlendirememektedir. Bu çalışmada eğitim seti olarak Inria hazır veri seti, test verisi olarak Göktürk-1 uydu verisinden yararlanılmıştır. Sınıflandırma için farklı optimizasyonlar (Adam, RMSprop, SGD, Adadelta, Adagrad, Nadam) ve çeşitli

parametreler (batch size ve öğrenme oranı) kullanılmıştır. Bina çıkarımında 3 farklı batch boyutu seçilmiştir. Bunlar;8,16 ve 32'dir. Öğrenme oranı olarak 2 farklı (1e-4 ve 1e-3) durum incelenmiştir. Yol ağlarında ise, sadece batch boyutları ve optimizasyon parametreleri irdelenmiştir. Çalışma kapsamında CNN algoritmalarından biri olan U-net mimarisi kullanılarak işlemler gerçekleştirilmiştir. Çalışmanın en temel amacı, Göktürk-1 uydu görüntü kullanılarak farklı bölgelerdeki bina ve yol ağlarının çıkarılmasının mümkün olduğunu göstermektir. Ayrıca modelde çeşitli parametrelerin değişimi durumunda sonuca olan etkisini incelemektir.

2. KAYNAK ARAŞTIRMASI

Uzaktan algılama sistemlerindeki yüksek konumsal çözünürlüklü uydu görüntüleri kullanılarak otomatik bina ve yol ağlarının sınıflandırılmasına yönelik çeşitli yöntemlerinin geliştirilmesi için büyük çaba sarf edilmiştir (Saralioglu ve Gungor, 2022). Literatürde yapılan incelemeler, zaman içinde sınıflandırma doğruluğunu artırmak adına geleneksel istatistiksel yöntemlerden başlayarak, güçlü derin öğrenme algoritmalarının kullanımına kadar geniş bir yelpazede ilerlediğini göstermektedir. Bu tez kapsamında sınıflandırma çalışmalarını iki temel kategoride incelenmiştir. Geleneksel yöntemler ve derin öğrenme yöntemleridir.

2.1. Bina Sınıflandırmasında Klasik Yöntemlerin Kullanılması

Bu zamana kadar yapılan birçok bilimsel makalede, uzaktan algılama görüntülerinden yararlanarak yapı/bina gibi nesnelerin tespit ve tanımlamasına odaklanılmıştır. Uzaktan algılama alanında kullanılan geleneksel yöntemler, veri türüne ve çözünürlüğe göre literatürde pek çok çeşitli yaklaşım bulunmaktadır. Geleneksel yöntemler, çoğunlukla piksel tabanlı, kenar algılama odaklı veya bölge tabanlı yaklaşımları içermektedir (Jiang ve ark., 2023).

Kenar algılama odaklı yöntemlerde, tipik olarak başlangıçta düşük seviyeli kenar algılama algoritmalarıyla çizgi segmentlerini belirlenir ve daha sonra çeşitli kurallara dayanarak bu segmentler bina kenarlarına gruplandırılır (Jaynes ve ark., 1994; Krishnamachari ve Chellappa, 1996; Lin ve ark., 1994; Turker ve Koc-San, 2015). Bölge tabanlı yaklaşımda belirli koşullar altında bina kenarları bina alanından türetilir. Bina alanlarını belirlemek için çeşitli sınıflandırma yöntemleri uygulanmaktadır (Ming ve ark., 2005; C. Tao ve ark., 2010; H. Wu ve ark., 2014).

Liow ve ekibi, bina tespiti için gölge kullanımında yenilikçi bir yaklaşım önerdi (Liow ve Pavlidis, 1990). Sonrasında yapılan araştırmalar, yüksek çözünürlüklü uzaktan algılama görüntülerinde gölge özelliklerine dayalı olarak binaları tanımlama ve çıkarma üzerine yoğunlaştı (Luo ve ark., 2015; Raju ve ark., 2014). Ayrıca, gölge ve bina arasındaki ilişki, yerel kontrastın artmasına neden olabileceği kanısına varıldı. Baltsavias (2004), nesne tespiti için mevcut bilginin çeşitli yönlerine odaklandı ve bu çalışma, nesne tespitinin pratik uygulamalarına yönelik önemli noktaları öne çıkardı. Brenner (2005), Binaların yeniden yapılanmasına büyük bir önem atfeden ve de görsel açıdan detaylı bir

inceleme sunan bir çalışma gerçekleştirildi. LIDAR temelli yeniden yapılandırma stratejileri ve bu stratejilerin temel başarıları da ele alındı. Gölge ve bina arasındaki ilişki durumu göz önünde bulundurularak, Pesaresi ve ark. (2008) ve Pesaresi and Gerhardinger (2010), PanTex yöntemi gibi gri seviye birlikte oluşum matrisi kontrast özelliklerini kullanan pratik çözüm önerdiler. Haala ve Kada (2010), bina yeniden yapılanması için havadan ve LIDAR yükseklik verilerini kullanarak yapılan yaklaşımları, aynı zamanda binaların cephe bilgilerini sağlayan verileri de inceleyen bir araştırma makalesi yayınlandı. Hu ve ark. (2011), bina tespiti için gölge, şekil, renk ve gölge çizgileri arasındaki açı benzerliği gibi çeşitli ipuçlarını kullanarak çalıştı.

Bu yöntemlere ek olacak şekilde birçok bilim adamı tarafından yardımcı bilgi tabanları yöntemi de kullanıldı. Bu yöntemin temelinde ise, segmentasyonu yapılacak nesnenin arka planında ya da yakın çevresindeki ortam karmaşıklığını çıkarmaktır. Özellikle kentsel alanlarda spektral karmaşıklık nedeniyle sorunların meydana geldiği görülmektedir. Bunun için gölge, stereoskopik hava görüntüsü veya dijital yükseklik modeli (DEM) verilerinden yararlanılmıştır (Lu ve ark., 2018). Ok (2013), farklı özelliklere sahip binaların VHR GeoEye-1 görüntüsü kullanılarak otomatik olarak tespit edilmesine yönelik bir çalışma gerçekleştirmişlerdir. Tuermer vd. (2013), çalışmasında kentsel alanlardaki nesneleri belirlemek için eğim histogramı (HOG- Yönlendirilmiş Gradyanlar Histogramı) yöntemini kullanmışlardır. Bu çalışmalara ek olarak, stereo bilgisi, bina bilgilerinin çıkarılmasında önemli kolaylıklar sağlayabilir düşüncesi ortaya atıldı (Siddiqui ve ark., 2016; Tian ve ark., 2013; B. Yang ve ark., 2016).

Geleneksel uzaktan algılama görüntü segmentasyon yöntemleri genellikle zayıf kalmaktadır. Kenar tespitine dayalı metotlarda, kapalı bölgelerin belirlenmesinde yaşanan zorluklar ve bölge temelli tekniklerin kenarları hassas bir şekilde ayıramama sorunları gibi çeşitli zorluklar mevcuttur(Shuyang Wang ve ark., 2020).

Artan uygulama talepleri ve büyüyen uzaktan algılama veri setleri, nesne tespiti ya da sınıflandırma konusunda daha yüksek beklentilere yol açmıştır (Jiang ve ark., 2023). Geleneksel görüntü segmentasyon yöntemlerinin, uzaktan algılama alanında yetersiz kaldığı görülmektedir. Yüksek çözünürlüklü uzaktan algılama görüntülerinin gelişimiyle beraber, farklı spektrumlar arasında değişen nesnelerin ve aynı spektruma sahip olan farklı nesnelerin segmentasyon doğruluğunu artırmak için derin öğrenme algoritmalarına olan ilgi artmıştır.

2.2. Bina Sınıflandırmasında Derin Öğrenme Algoritmalarının Kullanılması

Segmentasyon işleminde çeşitli teknikler kullanılmıştır, hala da bu teknikler geliştirilmeye çalışılmaktadır. Kenar algılamanın zengin geçmişinde, ilk kenar tanıma teknikleri genellikle gradyan ve yoğunluk bilgilerine dayanmaktaydı (Lu ve ark., 2018). Daha sonra araştırmacılar, yapay özellikler kullanarak kenarları belirlemeye yönelik tasarımlara geçti. Ancak, bu geleneksel yöntemler genellikle el ile belirlenmiş düşük seviyeli özelliklere dayandığı için doğrulukları garanti edilmesi zor ve çeşitli uygulamalara uyum sağlamakta zorlanıyordu. Ancak, yapay zekâ alanındaki hızlı ilerlemeler sayesinde, derin öğrenme yöntemleri doğal görüntü kenar tespiti konusunda mükemmel bir performans sergilemektedir. Günümüzde, klasik tekniklerin yerini giderek derin öğrenmeye dayalı yöntemler almaktadır. (Yu ve ark., 2007). Bu teknikle çok büyük veri kümelerinden hızlı ve otomatik bir şekilde çıkarım yapmak mümkündür. Derin öğrenmede segmentasyon uygulamaları ilk olarak sağlık alanında gerçekleştirilmiş olsa da sonrasında otonom araçlar, arttırılmış gerçeklik çalışmaları, arazi sınıflarının belirlenmesi gibi çeşitli alanlarda uygulamaları da bulunmaktadır (J. Gao ve ark., 2020; LeCun ve ark., 2015; Ronneberger ve ark., 2015; Sariturk ve Seker, 2023). Uydu görüntülerinden bina çıkarımı için derin öğrenme yöntemlerinden CNN algoritmaları ön plandadır ve araştırmalarda kayda değer ilerlemeler kaydetmişlerdir. Long ve ekibi, piksel düzeyinde sınıflandırmayı başaran ilk Tam Evrişimli Ağı (FCN) önerdi. Tamamen bağlı katmanın yerine karmaşık hesaplamalar içeren evrişim katmanını kullandılar ve orijinal boyutları geri yüklemek için ters evrişimi kullandılar. Bu kodlayıcı-kod çözücü mimarisi daha sonra anlamsal bölütlemede yaygın olarak kullanıldı(Long ve ark., 2015). Daha sonradan Ronneberger ve ark. kodlayıcı-kod çözücü mimarisinin geliştirilmiş hali olan U-net mimari yapısı geliştirmişlerdir (Ronneberger ve ark., 2015). Sağlık alanında kullanılmış ve görüntü segmentasyonunda iyi sonuçlar vermiştir. Araştırmacılar bu başarıdan sonra CNN algoritması altında çeşitli mimari yapıları geliştirmişlerdir. Wang ve ekibi, ResNet18 ön eğitim modeliyle birleştirilmiş iki kanallı bir görüntü özelliği çıkarma ağı geliştirdi. Ancak, ağın incelikli bölgelerin segmentasyonu üzerinde zayıf bir etkisi bulunmaktadır (E. Wang ve ark., 2019). Başka bir çalışmada, CU-Net, çoklu kısıtlamaları kullanarak ağın özellik çıkarım yeteneğini geliştirilerek ve çok ölçekli özellik temsili sağlanması hedeflendi (G. Wu ve ark., 2018). SiU-Net mimarisiyle, aynı görüntünün farklı ölçeklerini iki ağırlık paylaşımlı U-Net'e beslemekte ve ardından iki çıktı özellik haritasını birleştirmektedir. Bu yöntem sayesinde büyük binaların

doğruluğunu artmaktadır (Ji ve ark., 2018). MAP-Net, anlamsal bilgiyi çıkarmanın çoklu stratejisini benimsemektedir. Elde edilen özellik haritalarını farklı ölçeklerde birleştirdikten sonra, kanal dikkat modülü ve alan havuzu modülü kullanarak özellik kanalı ile özellik alanı arasındaki ilişkiyi yakalamaktadır (Zhu ve ark., 2020). Literatürde bu alan ile ilgili birçok çalışma bulunmaktadır.

Vakalopoulou ve ark. (2015), tarafından gerçekleştirilen çalışmada uydu görüntülerinden yararlanılarak bina tespitinde derin öğrenme algoritmasını kullanmışlardır. Sınıflandırma da derin öğrenme algoritması olarak AlexNet, makine öğrenme algoritması olarak DVM tercih edip, sonuçları karşılaştırmışlardır. Elde ettikleri bulgular doğrultusunda ortalama genel doğruluk oranının yaklaşık %90 olarak hesaplamışlardır. Ayrıca, yapay objelerin (insan tarafından yapılmış) tespit edilmesi ve izlenmesinin zor olduğunu belirtmişlerdir.

P. Zhang ve ark. (2016), yaptıkları çalışmada farklı konumsal çözünürlükteki uydu görüntülerinden yararlanarak arazi örtüsündeki değişimi incelemişlerdir. Derin öğrenme yöntemine dayalı bir mimari ağ kullanmışlardır. Piksel tabanlı olup, piksel komşuluğunun özelliklerinden yararlanılarak seyrek ve gürültü giderici otomatik kodlayıcı ile öğrenme işlemi gerçekleştirilmiştir. Elde ettikleri bulgular doğrultusunda farklı çözünürlükteki görüntüler arasında köprü kurmak için derin öğrenme yöntemlerinden yararlanılabileceğine varılmıştır.

Jun Zhang ve ark. (2019) yaptıkları çalışma ise, Hu ve ekibinin 2011 yılında yapmış oldukları çalışmaya benzemektedir. Zhang ve ekibi çalışmalarında yapay zekâ kullanmışlardır. Bu çalışmada, bina tespiti sürecinde renk, doku ve şekil özelliklerinin birleştirilerek daha etkili sınıflandırma sonuçları elde etmeyi amaçlamışlardır.

Maltezos ve ark. (2017), Almanya'daki Vaihingen'de üç alan ve Yunanistan'daki Perissa'da ise iki alan olmak üzere bitki örtüsüne ve zemine göre çeşitli bina verilerinin tespiti için ESA mimarisine dayalı bir derin öğrenme algoritması kullanılmışlardır. Buldukları sonuçları makine öğrenme algoritması olan destek vektörün doğrusal ve radyal tabanlı fonksiyonu ile karşılaştırmışladır. 5 alanın ortalaması alınarak elde edilen deneysel sonuçlara göre ESA mimarisi %81, doğrusal ve radyal tabanlı destek vektör makinesi için sırasıyla %76 ve %73'tir. ESA da her bir çalışma alanı için geçen sürenin 15 dakika olmasına karşılık, makine öğrenmesinde bu sürenin 1 dakikadan az olduğu görülmüştür. Bu durum, ESA'nın, DVM'e göre hesaplama karmaşıklığı açısından zayıf olsa da daha iyi bir işlemci ve daha fazla RAM kapasitesinin kullanılmasıyla, process (işleme) kısmını önemli ölçüde hızlandırabileceği sonucuna varılabilir.

H. L. Yang ve ark. (2017), hava fotoğraflarından yararlanarak Amerika Birleşik Devletlerindeki binaları tespit etmek amaçlı ESA mimarisinin 3 farklı algoritmasını kullanmışlardır. Bu algoritmalar; tamamen evrişimli sinir ağı (FCN), yinelenen sinir ağı (RNN) ve SegNet (Segmentasyon Ağı)'tir. Kullanılan hava fotoğrafları Ulusal Tarım Görüntüleme Programı (NAIP) tarafından toplanan 1 metrelik konumsal çözünürlüğe ve R, G, B, NIR olmak üzere 4 banttan oluşan görüntüleri içermektedir. 500×500 metrelik 5173 adet görüntüden, rastgele seçilen 4000 tanesi eğitim amaçlı, 1173 tanesi ise test amaçlı kullanılmıştır. Çalışma sonucunda en yüksek doğruluk %83,1 ile SegNet mimarisine aittir. 56 km^2 büyüklüğündeki bir bölge için model çıkarımı bir dakikadan az sürede gerçekleşmektedir. Bu durum sonuçların elde edilmesinde oldukça umut vericidir.

Lu ve ark. (2018), Massachusetts Bina Veri Kümesindeki görüntüleri kullanarak çıkarım işlemi gerçekleştirmişlerdir. Çalışmalarında zengin evrişim özellikli ağı RCF (richer convolution features network) kullanmışlardır. Veri arttırmak amacıyla görüntüleri 90,180 ve 270 derece döndürmüşlerdir. 750 × 750 piksel boyutlarında 1856 adet eğitim ve 56 test görüntüsü olmak üzere toplamda 1912 adet görüntü kullanmışladır. %89'luk F1 skor değerine ulaşılmıştır. Oluşturulan model ile bina kenarlarının tespit edilebilmesi mümkündür.

Gui ve ark. (2018), MBR-Net olarak adlandırılan çok dallı regresyon ağı ve U-Net mimarisini kullanmışlardır. Veri seti olarak 5000 × 5000 piksel boyutlarında, 0.3 m konumsal çözünürlüğe sahip ve ortorektifiye edilmiş Inria hava görüntüleri kullanılmıştır. Modellerde kullanılan eğitim sayısı 3000 ve 8000 olarak sonuçlar değerlendirilmiştir. Ve epok sayısı olarak 3000 görüntü kullanıldığında 25 epok ve 8000 eğitim kullanıldığın 10 epok olarak belirlemişlerdir. Elde ettikleri sonuçlar doğrultusunda, MBR-Net'in doğruluğu %50 ile %92,33 arasında, U-Net mimarisinin ise %50 ile %82 arasında değiştiğini tespit etmişlerdir. Önerilen MBR-Net yönteminin, U-Net'ten daha iyi performans gösterdiğini belirlemişlerdir.

Tun and Tun (2019), çalışmalarında DBN (Deep Belief Network / Derin İnanç Ağı-DİA) algoritmasını kullanmışlardır. Ücretsiz kullanıma açık olan Massachusetts bina veri seti yararlanarak çalışmalarını sürdürmüşlerdir. Elde ettikleri sonucu SegNet mimarisiyle karşılaştırmışlardır. Derin inanç ağından elde edilen doğruluğun, SegNet mimarisinden %4 oranında fazla olduğunu tespit etmişlerdir.

Tez çalışması kapsamındaki kaynak araştırmasına göre, derin öğrenme teknikleriyle bina tespiti, özellikle son on yılda sıkça tercih edilen güncel bir alan olarak görülmektedir. Tezin devamında kaynak araştırması sonucu elde edilen bilgiler Çizelge

2.1’de kullanılan mimari yapılar, eğitim verisinin boyutu, tercih edilen optimizasyon yöntemleri, epok, batch boyutu, öğrenme oranı ve elde edilen başarı oranları özetlenmiştir.

Çizelge 2.1. Literatürde bina çıkarımı için yapılan çalışmaların özeti

Veri/Uydu Görüntüsü	Derin Öğrenme Mimarisi	Eğitim Verisi Boyutu	Optimizasyon	Epok	BS	LR	Başarı Oranı	Alıntı
Ortofoto	ESA, Doğrusal DVM, Radyal Tabanlı DVM	10×10 piksel	*	*	*	*	ESA = %81 Doğrusal DVM=%76 Radyal Tabanlı DVM=%73	Maltezos ve ark. (2017)
Hava Fotoğrafı	FCN, RNN, SegNet	500×500 metre	*	120.000 iterasyon	3	*	FCN= %58.6 RNN = %80.5 SegNet = %83.1	H. L. Yang ve ark. (2017)
Massachusetts Bina Veri Kümesi	Zengin evrişim özellikli ağ	750 × 750 piksel	*	40,000 iterasyon	*	1e−7	%89	Lu ve ark. (2018)
Inria Hava Fotoğrafı	MBR-Net, U-Net	32 × 32 piksel	Adam	10 ve 25 epok	*	1e-4	MBR-Net= %50 ile %92,33 U-Net= %50 ile %82 arasında değişmektedir.	Gui ve ark. (2018)
Massachusetts Bina Veri Kümesi	Yama tabanlı derin inanç ağı, SegNet	64 × 64 piksel	*	10 epok	14		YTDİA=%94.5 SegNet=%88.9	Tun and Tun (2019)
WHU Hava Görüntüsü	RSR-Net	512 × 512 piksel	Adam	80 epok	6	1e-4	IoU = %88.32	Huang ve ark. (2021)
Inria veri,	CT-UNet	Inria	*	*	*	*	Inria da %85,33 IoU, WHU da %91,00	S. Liu ve ark. (2021)

WHU veri seti, Massachusetts veri seti		1024 × 1024, Mass. 384×384					IoU M veri setinde %83,92 F1 skoru	
Inria Hava Görüntüsü	BBU-Net	512 × 512	Adam	30 ve 80	5 ve 6	1e-4	%87,53 IoU ve %97,4 PA (piksel doğruluğu)	Z. Li ve ark. (2023)
Topcoder Urban3d Challenge veri seti, WHU Bina Veri Seti	HA U-Net	256×256	RAdam	80	16	--//	IoU= %74.96 F1 skor = %87.93	Xu ve ark. (2021)
Vaihingen seti ve WHU Binası veri seti	PSPNet, DeepLabV3, U-Net	256×256	*	70	16	1e-2	F1 skor, PSPNet =%85.80 DeepLabV3= %85.24, U-Net=%85.21	Xiang ve ark. (2021)
Inria Hava Görüntüsü	Xception U-Net	416×416	Adam	200	2	0,0004	Doğruluk, Xception=%95.14 U-Net=%93.73 F1 Skor, Xception=%82.14 U-Net=%81.44	She (2022)
Massachusetts veri seti, Inria hava görüntüsü	Building-A-Nets	56 × 256	RMSProp	*	*	1e-3	Iou = %78.73	X. Li ve ark. (2018)
WorldView-3	U-net-id	128 × 128	RMSprop	500	68	0,01	GD = %97.67 IoU= %58.2	Wagner ve ark. (2020)
LEVIR-CD veri seti	VGG-16 ResNet-50	256 ×256	RMSprop	200	64	1e-4	GD = %78.9	Mahmoud ve ark. (2021)
Dijital yüzey modelleri ve stereo verileri	ICT-Net	*	RMSProp	80	*	1e-4	IoU = %81	Dunaeva (2019)
Gaofen 2	TEA-8 U-Net		RMSProp	*	10	1e-4	F1 skor, TEA-8 =%80	Fu ve ark. (2019)

							U-Net=%77.9	
Wuhan Üniversitesi Bina Çıkarma Veri Seti, Massachusetts, Inria	ResNet-18, ResNet-50, MobileNetv2, Xception	300×300	SGD	*	Massac. 8, Inria 16, WHUBED 32	1e-3	Iou,3 veri Ortalaması, ResNet-18=%60.19 ResNet-50=%64.11 MobileNetv2= %52.24 Xception=%67.60	Atik ve ark. (2022)

*** ile ifade edilen kısımlar hakkında net bilgi ifade edilmemiştir.**

2.3. Yol Sınıflandırmasında Klasik Yöntemlerin Kullanılması

Uydu görüntüsü ya da havadan alınan bir görüntü üzerinden otomatik olarak yolları çıkarmak, akıllı trafik yönetimi, görüntü kayıt analizi ve topografik veri tabanı güncellemesi gibi çeşitli uzaktan algılama uygulamalarının temel birer bileşenidir (Guerrero-Ibañez ve ark., 2021; Mena, 2003; Tondewad ve Dale, 2020). Ek olarak, yol segmentasyon işlemi, araçlar, yapılar ve petrol kuyusu platformları gibi diğer nesnelerin tanımlanmasını önemli ölçüde etkileyen bir faktör olduğu için uzaktan algılama alanındaki araştırmacıların ilgisini çekmektedir (Hongjie He ve ark., 2022).

Geleneksel segmentasyon işlemi iki süreci içermektedir. Birinci aşama, görüntü üzerinden özellik çıkarımı; daha sonraki aşama ise görüntü piksellerinin sınıflandırılması işlemidir. Özellik çıkarma, manuel etiketleme yoluyla gerçekleştirilebilir; ancak bu yöntem araştırmacının uzmanlığına bağlı bir durumdur (Dai ve ark., 2019). Özelliklerin çıkarılması için görüntü üzerindeki geometrik, spektral doku, topolojik ve arka plan özelliklerinden yararlanılmaktadır (Vosselman ve De Knecht, 1995). Belirtilen bu dört özelliği çıkarmak için kullanılan çeşitli yöntemler bulunmaktadır. Bunlar; nesne tabanlı yöntem(Saba ve ark., 2016; Tan ve ark., 2008; Y. Wang ve ark., 2011; Yiğit ve Uysal, 2019), piksel tabanlı (Kalkan ve Maktav, 2010; Newman ve ark., 2011; Safaei ve ark., 2021), morfolojik filtreleme tekniği (Chaudhuri ve ark., 2012; Courtrai ve Lefèvre, 2016; Schubert ve ark., 2016; Y. Tao ve ark., 2022), görüntü eşleştirme(X. Lin ve ark., 2009; X. Lin ve ark., 2012; Jixian Zhang ve ark., 2011)'dir.

Geleneksel olarak, karayolu ağı bilgilerinin güncellenmesi genellikle saha araştırmalarına dayanır veya mobil haritalama sistemleri kullanılarak gerçekleştirilir. Ancak, bu yöntemler verimsiz, emek yoğun ve hızlı bir şekilde güncel karayolu ağı verilerini sağlamak zordur. Çünkü bir ülkenin tüm yol verilerini toplamak uzun zaman ve büyük bir çaba gerektirir. Bu nedenle son zamanlar uydu teknolojisi ve yapay zekâ yöntemlerinden yararlanılmaya başlanmıştır.

2.4. Yol Sınıflandırmasında Derin Öğrenme Algoritmalarının Kullanılması

Son on yılda, derin öğrenme tekniklerinin yaygınlaşmasıyla ve yüksek çözünürlüklü görüntülerinin daha iyi hale getirilmesiyle otomatik yol çıkarmanın zorluklarını aşmak için birçok araştırma yapılmıştır. Geleneksel yöntemlerde, yol

ağlarının tam hatlarıyla belirlenememesinin temelinde görüntünün arka planında ağaç, bina, gölge gibi yapıların bulunması etkilemektedir. Yapay zekâ ile bu problem çözülmeye çalışılmaktadır. Derin öğrenmeyle güçlendirilmiş temel klasik semantik segmentasyon algoritmaları şunlardır: FCN (Long ve ark., 2015), SegNet (Segmentasyon Ağı) (Badrinarayanan ve ark., 2017), U-Net (Ronneberger ve ark., 2015), DeepLab (Chen ve ark., 2018) ve PSPNet (Piramit Satır Sıralı Ağ)(Zhao ve ark., 2017).

Dai ve ark. (2019), yol tespiti için çok ölçekli bir derin artık evrişimli sinir ağı (MDRCNN) kullanarak farklı çözünürlüklere ve farklı sahnelere sahip iki veriyi birleştirmişlerdi. Yol bütünlüğünü artırmak için çeşitli tanımlayıcı tabanlı izleme bağlantıları kullandılar. Sherrah (2016) ise ImageNet veri setinden yararlanarak, yol alanını belirlemek için bir FCN'de evrişimsel katman ve alt örnekleme katmanı kullandı. Ancak, çok fazla FCL kullanılması durumunda ağın eğitim verimliliğini azaldığı sonucuna vardı. Eerapu ve ark. (2019), uzaktan algılama görüntülerine dayalı yol çıkarma görevlerinde sınıf dengesizliği sorunlarını hafifletmek için yoğun bir iyileştirme artık ağı (DRR-Net) önerdi. Önerilen mimarinin her bir modülünde, özellik öğrenimi için yalnızca kodlayıcı kısmında çeşitli ölçeklerde yoğun evrişim katmanlarının kullanılmasıdır. Ayrıca, her modülündeki artık bağlantılar, birleştirilmiş özellikleri sonraki DRR modüllerine yayarak rehberli öğrenme yolu sağlamayı amaçlıyordu. Y. Li ve ark. (2018), görüntüler üzerindeki karmaşık senaryoları elimine etmek amacıyla bir çalışma yaptılar ve çok ölçekli yolları etkili bir şekilde ele almak için hibrit bir CNN (HCN) modeli geliştirdiler. Bu model, yol ağlarının sıklık durumuna göre üç paralel omurgadan oluşmakta ve bunları birleştirerek doğru bir yol bölümlendirme sonucu üretmeyi hedeflemektedir. Y. Liu ve ark. (2018), yol yüzeylerini, kenarlarını ve merkez çizgilerini aynı anda çıkarmak için RoadNet adında çok görevli bir CNN modeli geliştirdi. Çeşitli çalışmalar yapılsa da yol ağlarını belirlemek için yapılan çalışmalarda daha çok kodlayıcı ve kod çözücü yapı benimsenmiştir. Birçok araştırmacı, görüntü üzerinden özellikleri tam olarak çıkarmak, çeşitli anlamsal bilgilerin birleşimini güçlendirmek ve gradyan kaybı ile aşırı uyum gibi sorunları ele almak için U-Net mimarisine dayanan çeşitli iyileştirmeler geliştirmişlerdir. Çizelge 2.2’de bu mimari yapıdan yararlanılarak yapılan çalışmalar bulunmaktadır. Performansı artırmaya yönelik geliştirmeler arasında, hesaplama hızının ve bellek kullanımının iyileştirilmesi bulunurken, kodlayıcı ve kod çözücü yapısında iyileştirmelerde yapılmıştır. Yapılan bu gelişimler modelin performansını, ağların hassasiyetini ve doğruluğuna bir miktar olumlu katkı sağlamışlardır (Ronneberger ve ark., 2015).

Çizelge 2.2 Literatürde yol çıkarımı için yapılan çalışmaların özeti

Veri/Uydu Görüntüsü	Derin Öğrenme Mimarisi	Eğitim Verisi Boyutu	Optimizasyon	Epok	BS	LR	Başarı Oranı	Alıntı
Massachusetts yol veri seti	atrous uzamsal piramit havuzlamayı (ASPP)	512 × 512	Adam	50	*	1e-4	F1 skoru %83.5	Hao He ve ark. (2019)
DeepGlobe veri seti	LinkNet, D-LinkNet AD-LinkNet	*	Adam ve RMSprop	*	2 ile 8	1e-4	IoU= LinkNet = %63 D-LinkNet=%64.12 AD-LinkNet = %78.5	M. Wu ve ark. (2019)
KITTI Vision Benchmark Suite ve Cityscapes veri seti	MVS-CNN	*	SGD ve RMSprop	50	*	*	IoU= %97.1	Junaid ve ark. (2020)
İHA	U-Net, DeepLab v3+, PSPNet	*	Adam	50/100	2	1e-4	IoU, U-Net = %49.26 DeepLab v3+=%70.33 PSPNet=%72.86	Hao ve ark. (2023)
Toronto yol veriseti	U-Net, FCN	256 x 256, 512 x 512	Adam, SGD	100, 80	*	1e-4, 1e-2	256 x 256, U-Net = %96.33 FCN=%90.23 512 x 512, U-Net = %96.18 FCN=%97.69	Ozturk ve ark. (2020)
Oxford RobotCar veri kümesi	RCNet	*	SGD, Adadelta, RMSprop, Adagrad, Adam, Nadam,	500	32	1e-3	Adam Doğruluk=%99.90	Dewangan and Sahu (2021)

			Adamax					
Deepglobe veri seti ve Massachusetts veri seti	RoadFormer,	1024 × 1024	Adam	*	4, 2	2e-4	IoU=%70.86	X. Liu ve ark. (2023)
Deepglobe veri seti ve Massachusetts veri seti	LDANet,	256 × 256	Adam	*	*	1e-3	IoU = %76.21	B. Liu ve ark. (2023)
Hava görüntüsü	DeepLabV3+	1024 × 1024	Adam	3 farklı epok Kullanılmış *	4	8e−5	Doğruluk = %86.83	Lourenço ve ark. (2023)
Havadan görüntüsü	CFCN	256 × 256	Adam	20	*	1e-4	Doğruluk = %83,6	Guo ve ark. (2022)
Sentinel-2A	ResUnet, ResUnet-a, U-Net, Attention U-Net, LinkNet, Dilated-LinkNet	128x128	Adam	100	32	1e-3	IoU, ResUnet=%56.60 ResUnet-a=%56.25, U-Net=%57.33, Attention U-Net=%57.78 LinkNet=%56.23, Dilated-LinkNet=%56.35	Gupta ve ark. (2022)
Dubai'nin MBRSC uydu görüntüleri	U-Net	*	Adam	50	8	1e-3	Dice = %87	Patil ve ark. (2022)

*** ile ifade edilen kısımlar hakkında net bilgi ifade edilmemiştir**

3. MATERYAL VE YÖNTEM

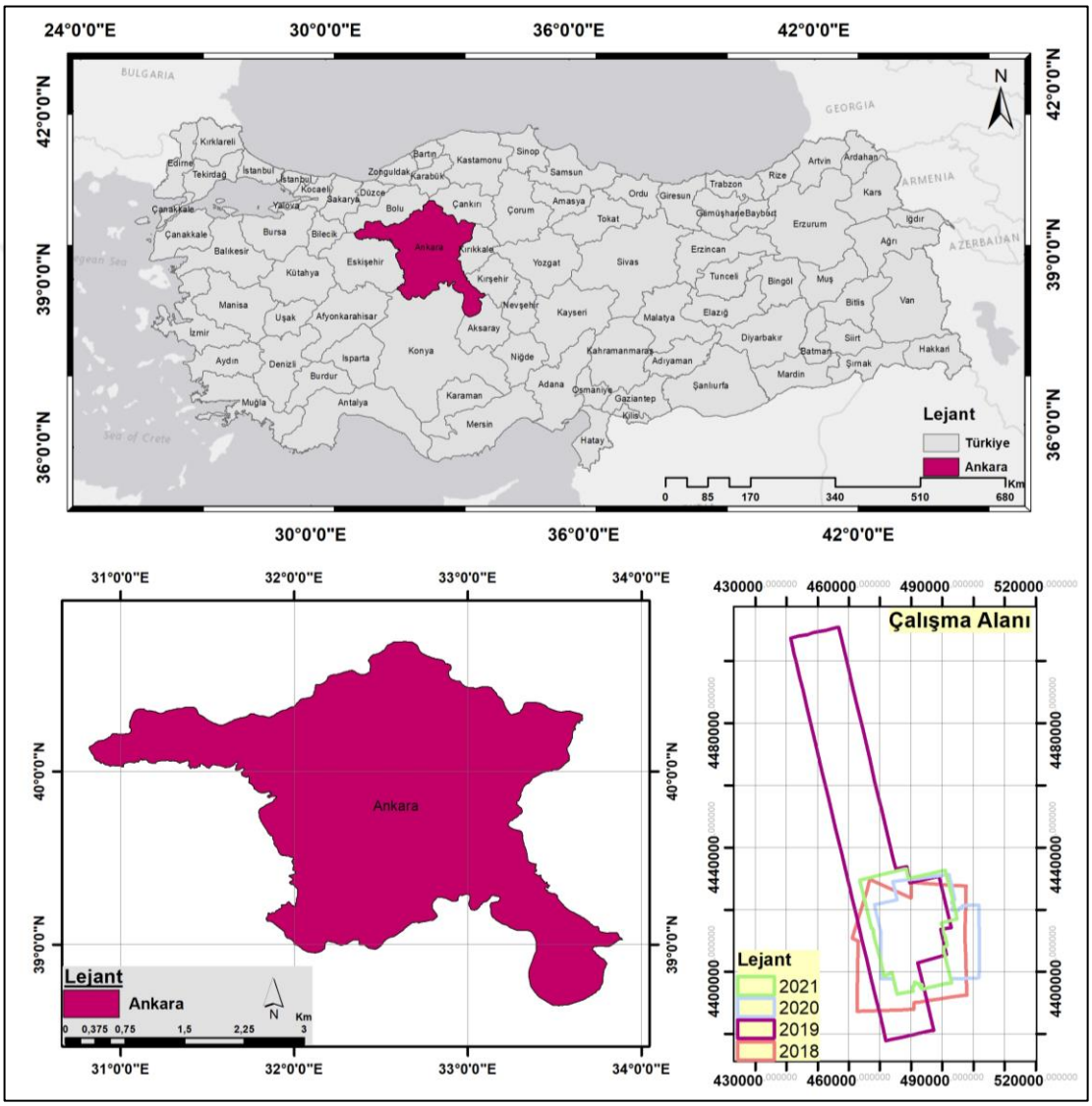
Bu bölümde, çalışma alanı olarak seçilen Ankara ili detaylı olarak tanıtılacaktır. Eğitim amaçlı kullanılan Inria veri setine ve test amaçlı kullanılan Göktürk-1 verisine ait ayrıntılı bilgiler sunulacaktır. Ek olarak, test verilerinin toplanmasında kullanılan Göktürk-1 uydusu hakkında teknik bilgiler verilecek ve verilerin bant özelliklerine değinilecektir.

3.1. Çalışma Alanı

Bu tez çalışmasında 39.57º K enlem ve 32.53º D boylamlarında yer alan Ankara ilinin belirli bölgeleri kullanılmıştır (Şekil 3.1). Ankara, köklü bir tarihe sahip bir şehirdir. Zaman içerisinde Anadolu'da bir yerleşim merkezi olarak ve doğu ile batı arasında bir bağlantı noktası olarak önem kazanmıştır(Mikaeili, 2016). İç Anadolu bölgesinde yer alan Ankara, kentlerin ortasında ticari yolların ana ulaşım bağlantıları üzerinde yer almaktadır (Akçura, 1971). Şehrin doğusunda Kırıkkale, batısında Eskişehir, güneyinde Konya bulunmaktadır. Ayrıca kuzey doğusunda Çankırı, kuzey batısında Bolu ve güney doğusunda Kırşehir ve Aksaray illeriyle sınırlarını paylaşmaktadır. Şehri çevreleyen çevre yolu, ilçeler arası bağlantıyı kolaylaştırarak ve iller arası ulaşımı sağlayarak şehir içi trafiği düzenlemektedir. Ekonomik aktivite cumhuriyetin ilk yıllarında tarım ve hayvancılığa dayanıyorken, günümüzde büyük ölçüde ticaret ve endüstriye dayanmaktadır. Şehir topraklarının yarısı hâlâ tarım amaçlı kullanılsa dahi, tarım ve hayvancılığın ağırlığı gün geçtikçe azalmaktadır. Ankara ili, 850 ila 1000 metre aralığında ovalar, 1000 ila 1400 metre aralığında yaylalar ve 1400 metre üzerindeki yüksek kesimlerde tepeler ve dağlarla karakterizedir. Hatip Ovası, Mürted Ovası ve Çubuk Ovası Ankara ilinin yapı taşlarındandır. Şehrin geneline bakıldığında 2000 metre yüksekliği geçen dağ bulunmamaktadır(Okan, 2023).

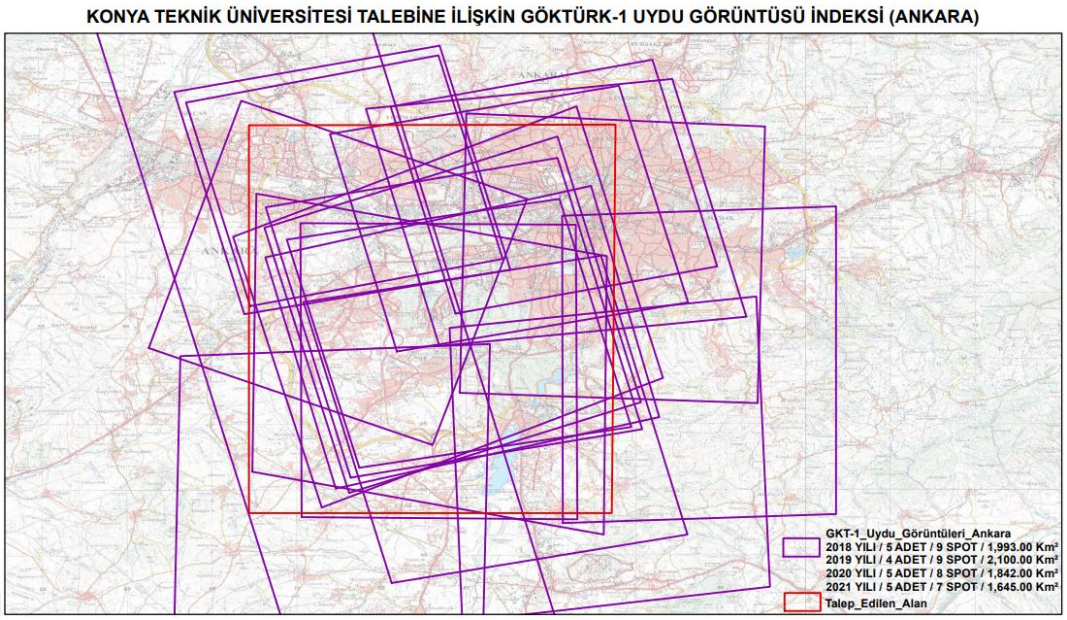
13 Ekim 1923'te kabul edilen yasayla, Türkiye Cumhuriyeti'nin başkenti Ankara olmuştur, böylece şehir yeni Türk devletinin merkezi haline gelmiştir. Şehir cumhuriyetin ilan edilmesinden itibaren sürekli göç almış ve gecekondu yapılaşmasını yaşayan ilk kent durumundadır. Ankara'da, insanlar henüz planlanmamış hazine arazilerini seçerek, kırsal alanlardaki geleneksel yapı tekniklerini kullanıp, kendi yaşam tarzlarına uygun tek katlı konutlar yapmaya başladıklarında, ilk gecekondu bölgeleri oluşmaya başlamıştır(Çam, 2023). Şehrin alansal yayılması oldukça hızlı olmuştur. Bayar (2020) yapmış olduğu

çalışmada, 1924 yılında yaklaşık 2 km^2'lik bir alanda başlayan Ankara şehri, 1955'te 43 km^2'ye, 1985'te 279 km^2'ye ve 2018'de ise 468 km^2'ye kadar genişlemiştir. Şehrin bu şekilde hızlı genişlemesi potansiyel sosyal, ekonomik ve çevresel sorunlarını beraberinde getirmektedir. 1950 yılından itibaren göç alan şehrin günümüzde nüfusu, 5.803.482 kişidir. Ankara ilinin ilçelerine göre nüfusu incelendiğinde, en büyük nüfusa sahip ilçe Çankaya iken, en az nüfusa sahip olan ilçe ise Evren'dir (Batuman, 2013).



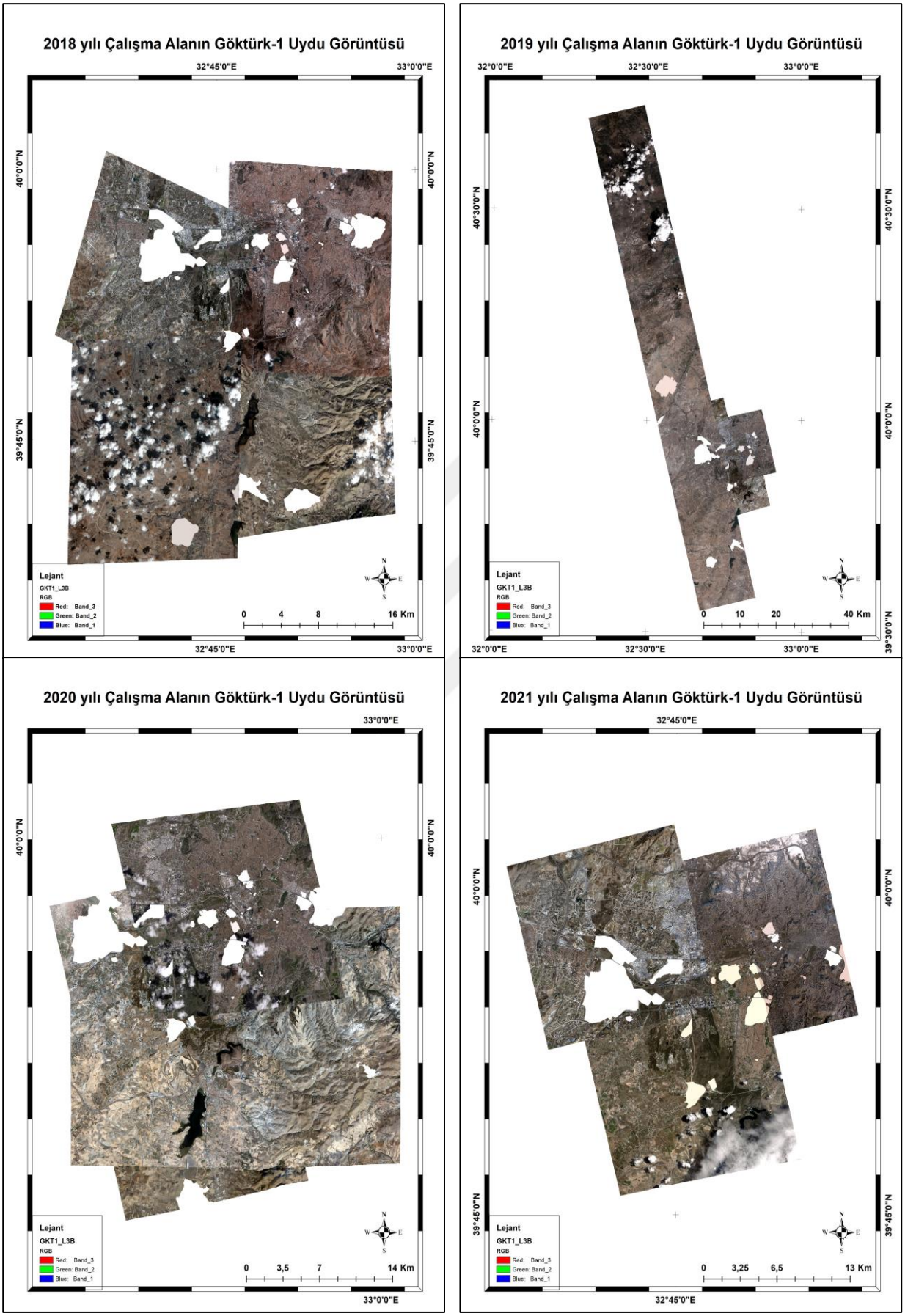
Şekil 3.1. Ankara ilinin Türkiye'deki konumu ve çalışma alanına ait veri yılları

Çalışmada kullanılan Göktürk-1 uydu görüntüsüne ait veriler Harita Genel Müdürlüğü'nden (HGM) temin edilmiştir. Şekil 3.2'de çalışma bölgesi için talep edilen alan kırmızı ile ifade edilirken, 2018, 2019, 2020 ve 2021 yıllarında bulunan uydu görüntülerini mor renk ile gösterilmiştir.



Şekil 3.2. HGM veri talebi sonucu Göktürk-1 uydu görüntü indeksi

Şekil 3.3'te temin edilen görüntülerden her bir yılın kapladığı alan ifade edilirken, bu görüntülere ait ayrıntılı bilgiler ve bilgilerin elde edilmesi için yazılan kodlar EK'te sunulmuştur. Bu görüntülerden bazılarının hava şartlarından dolayı bulutlu olması, yasak ve askeri bölgeler içermesinden ötürü, çalışma alanı için ortak bir alan belirlenmiştir. Ayrıca bazı görüntülerde kayıklıkların meydana geldiği kısımlardan dolayı yüksek katlı binaların yan cephelerinin gözükmesi gibi durumlarda söz konusudur.



Şekil 3.3. Ankara iline ait 2018,2019, 2020 ve 2021 yılı Göktürk-1 uydu görüntüleri

3.2. Uygulamada Kullanılan Veri Setleri

Çalışmada iki veri setinden yararlanılmıştır. Bunlardan bir tanesi Göktürk-1 uydu görüntüsü, diğeri ise Inria hava fotoğrafıdır.

3.2.1. Göktürk-1 uydu görüntüsü

5 Aralık 2016 tarihinde fırlatılan Göktürk-1 uydusu ve 681 kilometre yükseklikte güneş eş zamanlı bir yörüngeye yerleştirilmiştir. Bu yörünge, uydunun her gün aynı saatte ekvatoru geçmesini ve Dünya'nın aynı bölgelerini sürekli olarak görüntülemesini sağlar (Arasan ve ark., 2020). Türk Havacılık ve Uzay Sanayii A.Ş. (TUSAŞ) tarafından Türkiye'de geliştirildi. Ancak projede uluslararası iş birlikleri de önemli bir rol oynadı. Uydunun optik kamerası, İtalyan Telespazio ve Fransa Thales Alenia Space firmaları tarafından üretilmiştir. Uydu, yüksek çözünürlüklü optik kameralarıyla dünyanın çeşitli bölgelerini detaylı bir şekilde gözlemlemek için tasarlanmıştır. Göktürk-1'in fırlatma işlemi, yüksek teknolojiye sahip bir fırlatma aracı kullanılarak gerçekleştirildi ve Türkiye'nin ilk yüksek çözünürlüklü görüntüleme özelliğine sahip yer gözlem uydusudur (Aytekin ve Topan, 2022; Ünal ve Yıldız, 2021). Türkiye'nin uzay teknolojisinde önemli bir ilerleme kaydettiğini göstererek, stratejik ve ticari açıdan önemli bir başarıyı temsil etmektedir. Uydu ilk olarak askeri istihbarat amaçlı tasarlanmış olsa da uydudan elde edilen görüntüler, birçok farklı alanda kullanılmaktadır. Haritacılık, şehir planlama, ormancılık, afet yönetimi gibi birçok sivil alanda da Göktürk-1 uydu görüntülerinden yararlanılmaktadır (Aytekin ve Topan, 2022). Çizelge3.1'de uyduya ait teknik özellikler detaylı bir şekilde listelenmektedir. Göktürk-1'in geniş alan, şerit ve stereo gibi farklı görüntüleme modlarına sahiptir. 15 kilometre genişliğinde bir alanı tek seferde görüntüleyebilir. Zaman çözünürlüğü yaklaşık 2.5 gündür. Bu durum bir bölgeden geçtikten sonra aynı bölgeyi tekrar görüntüleyebildiği zaman aralığını ifade etmektedir (Arasan ve ark., 2020). X-Bandı ve S-Bandı haberleşme bantları, uydunun yer istasyonlarıyla yüksek hızlı veri aktarımı yapmasını sağlar. Bu, gerçek zamanlı görüntüleme ve acil durum durumlarında hızlı yanıt verme yeteneği sağlar. Ayrıca, Göktürk-1'in tasarım ömrü yaklaşık 7 yıl olarak planlanmıştır. Bu durum, uzun vadeli görevlerde güvenilir bir veri kaynağı olarak kullanılabileceğini göstermektedir. Nokta, Şerit, Geniş Alan, Stereo gibi görüntüleme modları, farklı ihtiyaçlara göre özelleştirilmiş görüntüleme seçenekleri sunar. Özellikle stereo görüntüleme, 3D modelleme ve topografik analizler için önemlidir.

Çizelge 3.1. Göktürk-1 Uydusunun Teknik Özellikleri

Göktürk-1 Uydusunun Teknik Özellikleri	
Fırlatılma Zamanı	05 Aralık 2016
Ekvatordan Geçiş Zamanı	10:30 (Yerel saat yükselen düğüm)
Faydalı Yük	Elektro-Optik Kamera
Yörünge	~681 km (Güneş Eş zamanlı)
Optik Sistem	Korshe Tipi Teleskop
	Odak Uzaklığı = 18140 mm
Çözünürlük	< 1m
Tekrar Ziyaret Etme Zamanı	~2.5 gün
Spektral Bantları	Pankromatik, RGB, Yakın Kızılötesi
Tasarım Ömrü	~7 yıl
Kütle	~1000 kg
Görüntüleme Modları	Nokta, Şerit, Geniş Alan, Stereo
Haberleşme Bantları	X – Bandı / 620 Mbps
	S – Bandı / 1,6 Mbps
Günlük Görüntü Kaydetme Kapasitesi	902 Tek Görüntü
En faza Bindirmeli Görüntü Genişliği	410 km Tek Geçiş
Yalpa/Yunus Açısı	$\pm 45°/\pm 30°$
Görüntü Boyutu	15 km (Nadir)

Çizelge 3.2'de ise, Göktürk-1 uydusunun görüntüleme bantlarının özelliklerini özetlemektedir.

Çizelge 3.2. Göktürk-1 uydusu bant özellikleri

Göktürk 1 Uydusu Bantlarının Özellikleri	
Geometrik Çözünürlük	Pankromatik 0.5m
	Çok Bantlı 2m
Zamansal Çözünürlük	~2.5 gün
Radyometrik Çözünürlük	12 bit
Tayfsal Bandlar (μm)	Pankromatik
	Mavi
	Yeşil
	Kırmızı
	Yakın Kızılötesi
Dedektör Boyutu	Pankromatik 6 dizi, toplamda 31600 piksel
	Çok bantlı 6 dizi, toplamda 7900 piksel

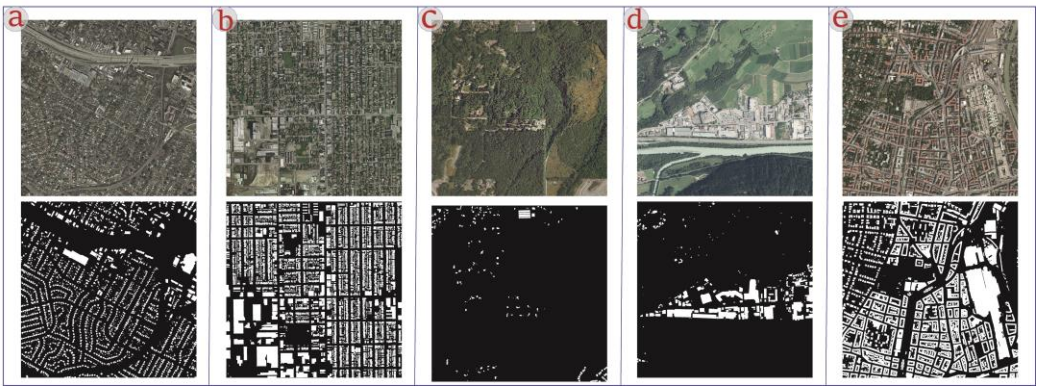
Göktürk-1 uydusunun görüntüleme bantları, geniş bir yelpazedeki uygulamalar için uygun olan yüksek kaliteli görüntüler üretmek için tasarlanmıştır. Uydu pankromatik, mavi, yeşil, kırmızı ve yakın kızılötesi bantlarda görüntüleme yaparak uydunun farklı türdeki arazileri ve nesneleri ayırt edebilmesini sağlar. Geometrik çözünürlük bir pikselin arazi üzerindeki gerçek boyutunu, radyometrik çözünürlük ise sensörün her bir pikselde ölçtüğü ışık miktarını ifade etmektedir. Kısacası gelen ışık miktarını kaç farklı seviye ya da değere ayırabildiği ile ilişkili bir kavramdır. Uydunun pankromatik bandı 0.5 metrelik

bir çözünürlüğe sahipken, çok bantlı bandı 2 metrelik bir çözünürlüğe sahiptir. Bu durum, pankromatik bandın daha ayrıntılı görüntüler üretebileceği anlamına gelir. Göktürk-1'in 12 bitlik bir radyometrik çözünürlüğü olması sayesinde geniş bir parlaklık aralığını ayırt edebilmektedir (Ünal ve Yıldız, 2021).

3.2.2. Inria veri seti

Çalışmada kullanılan ilk veri kümesi, Inria Hava Görüntüsü Etiketleme Veri Seti olarak seçilmiştir. Bu veri seti, Fransız Ulusal Bilgi ve Otomasyon Enstitüsü tarafından 2018 yılında yayımlanmıştır (She, 2022). Amerika Birleşik Devletleri (Austin, Chicago, Kitsap) ve Avusturya (Tyrol, Vienna) da bulunan kentsel bölgeleri içermektedir. Bölgeler nüfusun fazla olduğu kentsel alan ve nüfusun az olduğu kırsal alanları içermektedir. (Maggiori ve ark., 2017). Veri seti çeşitli bina tiplerine ait görüntülerin farklı coğrafi konumlardan seçilmesi, bu veri kümesini kullanan modellerin genelleme yeteneğini artırmaktadır (Maggiori ve ark., 2017; Sariturk ve Seker, 2023). Veri seti 810 km^2'lik bir alanı kapsamaktadır (Sariturk ve Seker, 2023). 0.3m konumsal çözünürlüğe ve 3 bantlı RGB görüntüleri içeren eğitim setindeki her görüntü bina ve bina dışı sınıf olarak ikiye ayrılmıştır. 5 farklı şehrin her birinden 36'şar görüntü olmak üzere toplamda 180 adet 5000×5000 boyutlarında RGB ve bunlara ait maskelenmiş görüntü bulunmaktadır. Birbirleriyle örtüşmeyecek şekilde ayarlanmış her bir görüntünün kapladığı alan 1500 m x 1500 m'dir (Maggiori ve ark., 2017).

Şekil 3.4' de 5000×5000 boyutlarında beş şehirden her birine ait, RGB ve maskelenmiş görüntüler bulunmaktadır.

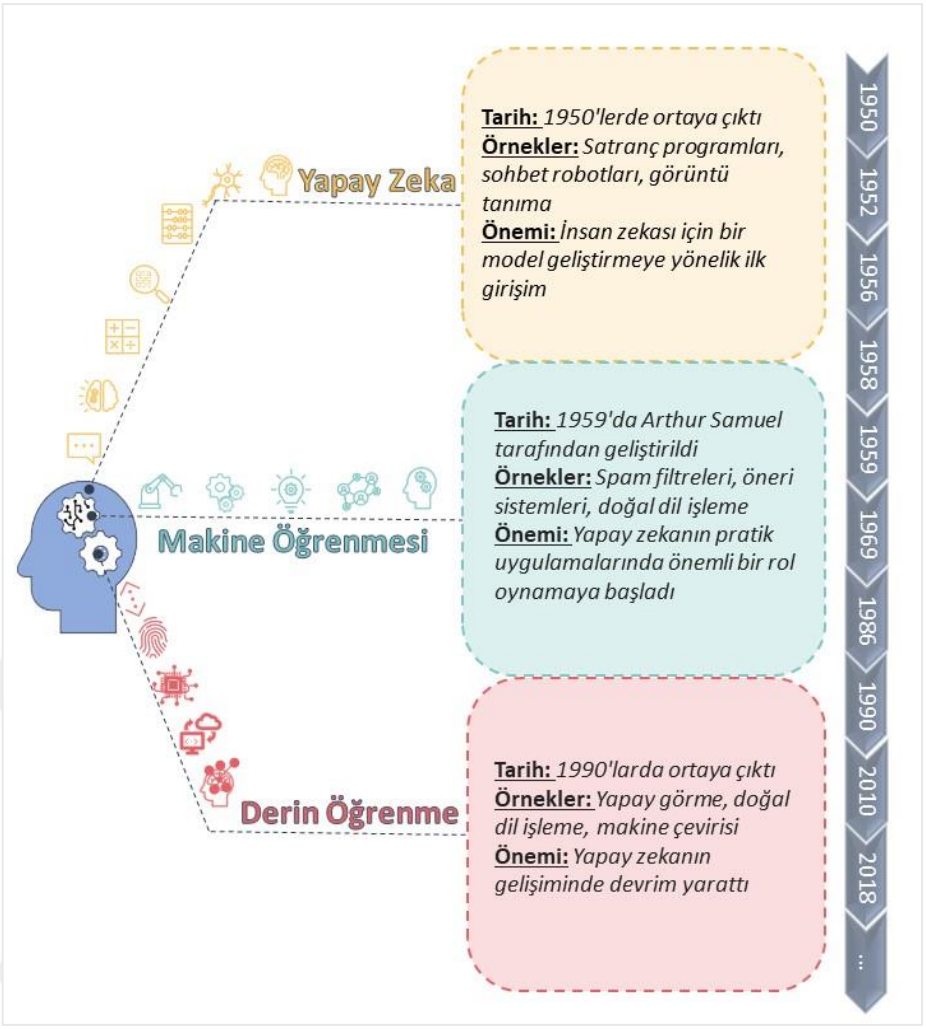


Şekil 3.4. Inria Hava Görüntüsü Etiketleme Veri Kümesinden RGB görüntüsü ve ilgili maske
a) Austin, b) Chicago, c) Kitsap, d) Tyrol ve e) Vienna

3.3. Yapay Zekâ ve Tarihçesi

Yapay zekanın bulunması 1950'li yıllar olarak gözükse de sıfır rakamının bulunmasıyla algoritmanın temelini atan El-Harezmî'ye kadar dayanmaktadır(Özgür, 2021). İkili sistem üzerine kurulan bilgisayar programcılığının ilk adımı olarak düşünülebilir. 1900'lü yıllarda bilim insanı olan Vannevar Bush "Makinelerin gelecekte düşünebileceğini" söyleyerek bu alanda çalışmalar yürütmüştür. Douglas Engelbart ise "İnsan aklının arttırılma" isimli makalesiyle arttırılmış zekâ kavramının dile getiren kişidir. 1950'li yıllara gelindiğinde Alan Turing makinelerin öğrenmeyi öğrenmesi ve evrimleşmesi üzerine çalışmalar yaparak yapay zekanın ilk adımlarını atan kişi olarak kabul edilir. Turing makinelerin karmaşık programları çalıştırabileceğini dile getirmiştir (Buchanan, 2005). 1954 yılında, Amerikalı matematik profesörü McCarthy, bilgisayarların zekasını tanımlamak için ilk kez "yapay zeka" kavramını kullanmıştır. McCarty ve birkaç bilim insanın ortak fikrine göre bilgisayarlar ile insan zihnini taklit etmek mümkün olabilirdi. Bu amaçla, 1956 yılında beyni nasıl simüle edeceğini tartışmak amacıyla bir çalıştay düzenlemişlerdir. Birçok bilim insanı Dartmouth Yaz Araştırma Projesinde bir araya gelmiştir(McCarthy ve ark., 2006). John von Neumann, boş tüpler ve telgraf röleleri kullanarak temel sinir hücresi işlevlerini taklit etmeye çalıştı(Muehlenbein, 2014). Bu dönemde, yaklaşık olarak 1956-1957 yıllarında, Frank Rosenblatt ve nörobiyolog Charles Wightman, ilk başarılı sinir ağı bilgisayarı olan Mark I Perceptron'u geliştirdiler (Hay ve ark., 1960; Kanal, 2003). Başarılı işlemler yapılsa dahi yeterli bütçenin ve hayata geçirebilecek teknolojinin olmamasından dolayı çalışmalar 1990'li yılların başına kadar duraklama dönemine girdi. Bu dönemde gerçekleştirilen çalışmalara "sembolik yapay zekâ" olarak isimlendirilmiştir. Bu döneme ise "yapay zeka kışı" demişlerdir (Buchanan, 2005; Van de Sande ve ark., 2022). Fakat, John Hopfield tarafından Hopfield ağları olarak isimlendirilen ağ yapısı icat edildi ve Teuvo Kohonen ise, kendi kendini düzenleyen özellik haritalarını tanımladı(de Carvalho ve ark., 2020; Hopfield ve Tank, 1985). Bu kişiler sayesinde yapay zekâ konuları tekrar gündeme geldi ve 1987'yılında Elektrik ve Elektronik Mühendisleri Enstitüsü (IEEE) tarafından yapılan konferansa 1800 kişiden fazla katılım sağlandı.

Yapay zekâ, insan zihninin yaptığı düşünsel faaliyetleri otomatikleştirme sürecidir olarak tanımlanabilir (Şennur, 2022). Şekil3.5'te yapay zekâ, makine öğrenmesi ve derin öğrenmenin kronolojik tarihçesi ifade edilmektedir.

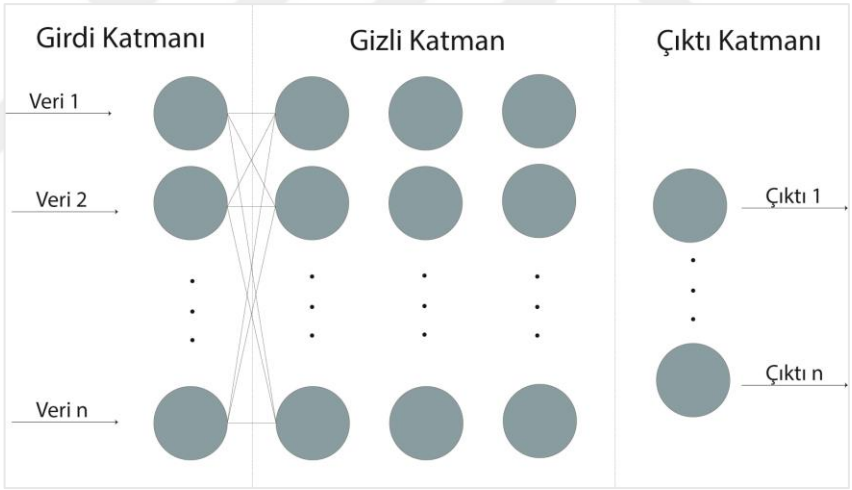


Şekil 3.5. Yapay zekanın tarihsel süreci

Derin öğrenme, makine öğrenmesinin alt dalı ve derin bir sinir ağı modelidir (Kırcalı, 2022; Şennur, 2022; Zheng ve ark., 2020). Derin öğrenme ve makine öğrenmesi kavramları birbirine sıkça karıştırılmaktadır. Fakat iki algoritmayı birbirinden ayıran temel özellik, öğrenme şekilleridir. Derin öğrenmede belirli kuralların kodlanması yerine verilerin öznitelik bilgilerini çıkararak kendiliğinden öğrenebilmesi söz konusudur(Kırcalı, 2022). Ayrıca derin öğrenmede girdi verisi bir bütün olarak ele alınmaktadır. Bu sayede bilgi kayıpları engellenmiş olur (Şennur, 2022).

Derin öğrenme, geniş veri setlerinden hızlı ve otomatik bir şekilde görüntü özellikleri çıkarabilir ve karmaşık modelleri yinelemeli bir şekilde kullanarak regresyon algoritmalarının doğruluğunu artırabilir. Regresyon yöntemleri, istatistiksel girdi dosyalarının özellikleriyle ilişkili beklenen çıktıyı tahmin eder. Standart bir prosedür, programın test aşamasının özelliklerine dayalı olarak bir model oluşturması veya bu modeli yeni bilgilerin etkilerini tahmin etmek için kullanması olarak ifade edilebilir

(Jiang ve ark., 2023). Derin öğrenme genel yapısı itibariyle üç katmandan oluşur: giriş katmanı, gizli katman ve çıkış katmanıdır (Şekil3.6). Veriler, giriş katmanına girer. Ağ, dış kaynaktan gelen verileri bu giriş katmanında tutar. Bu katmandaki her bir düğüm, veri setindeki özelliklere tekabül eder. Giriş katmanındaki veriler, gizli katmandaki düğümlere ağırlıklı bağlantılar aracılığıyla iletilir (Önün, 2019). Veriler, giriş katmanından sonra gizli katman ya da katmanlara yönlendirilir (Erhandı, 2020). Bu katman, hesaplama süresi ve karmaşıklığı gibi etkenleri belirleyebilir. Uygulamaya bağlı olarak, kullanılan modele göre ara katmanın sayısı değişebilir. Ara katmanda, veriler ağırlıklar kullanılarak işlenir ve çıktı katmanına iletilir. Çıktı katmanı ise sonuçların bulunduğu bölümdür. Ara katmandan gelen bilgiler, çıktı değerlerini bu kısımda oluşturur. Buradaki hata fonksiyonları, beklenen sonuçlar ile elde edilen sonuçlar arasındaki farkı hesaplar. Optimizasyon fonksiyonları ise hesaplanan hata değerini kullanarak her bir ağırlığın güncellenmesini sağlar, böylece öğrenme gerçekleştirilir (Arıkan ve Yıldız, 2023; Erhandı, 2020; Önün, 2019).



Şekil 3.6. Derin öğrenmenin genel yapısı

Derin öğrenme, çeşitli alanlarda sıklıkla kullanılmaktadır. Bilgisayarlı görme, konuşma tanıma, nesne algılama, doğal dil işleme ve diğer alanlarda yaygın olarak kullanılmış ve iyi sonuçlar elde edilmiştir (Kırcalı, 2022; Krizhevsky ve ark., 2017; N. Zhang ve ark., 2018; Zheng ve ark., 2020).

Derin öğrenme algoritmaları içerisinde evrişimsel sinir ağları, derin inanç ağları, tekrarlayan sinir ağları, Uzun-Kısa Süreli Hafıza, Kısıtlı Boltzmann Makineleri, Derin Oto-Kodlayıcılar (Auto Encoder) olacak şekilde farklı ağ modelleri bulunmaktadır. Bu

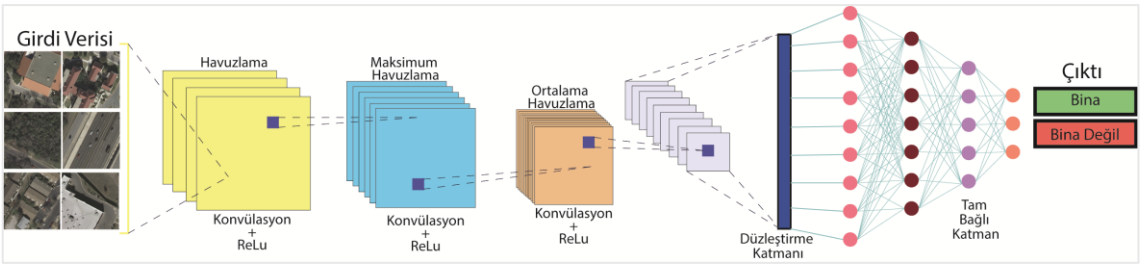
ağlar, ihtiyaca yönelik donanım kapasitesinin artması sonucunda katman ve bağlantı sayısı değiştirilerek, farklı isimler altında yapılan değişiklerle oluşturulmuştur(Metlek ve Çetiner, 2021; Şennur, 2022).

Görüntü anlamsal bölümleme yöntemleri, derin öğrenme temelli yaklaşımların artan ilgi ve araştırma konusu haline gelmesiyle birlikte evrişimli sinir ağı teknolojisinin gelişimine önemli katkılarda bulunmuştur(Jiang ve ark., 2023).

3.3.1. Evrişimli sinir ağları ve temel kavramlar

Derin öğrenme yöntemleri arasında görüntü ve nesne sınıflandırmak amaçlı en yaygın kullanılan Evrişimsel Sinir Ağlarıdır. LeCun ve ark. 1989 yılında karakter (posta kodu, basit basamak) tanımak amacıyla evrişim sinir ağını önermişler ve LeNet olarak isimlendirmişlerdir (Krizhevsky ve ark., 2017). 2012'deki ImageNet Large Scale Visual Recognition Challenge (ILSVRC) yarışmasında AlexNet ortaya çıktı. 2014'te ise GoogleNet, Karen Simonyan ve Andrew Zisserman'ın geliştirdiği VGGNet, ve Kaiming He ve ekibinin ResNet'i gibi evrişimsel sinir ağı mimarileri ön plana çıktı ve oldukça yaygın bir şekilde kullanılmıştır. Şekil3.7'de Evrişimsel sinir ağlarının genel yapısı ifade edilmiştir. SegNet ve U-Net gibi, bilinen iki önemli kodlayıcı ve kod çözücü ağı mevcuttur, bu ağlar genellikle görsel segmentasyon görevlerinde kullanılır ve yoğun bir şekilde araştırılmış ve geliştirilmiştir (Badrinarayanan ve ark., 2017; Ronneberger ve ark., 2015).

Her ne kadar pek çok bilgisayarlı görme görevi derin evrişimli sinir ağlarıyla başarılı bir şekilde gerçekleştirilmiş olsa da, uydu görüntüsüyle bina veya yol sınıflandırmasında CNN'lerin kullanımı nispeten yenidir (Saralioglu ve Gungor, 2022).



Şekil 3.7. CNN mimarisinin genel yapısı

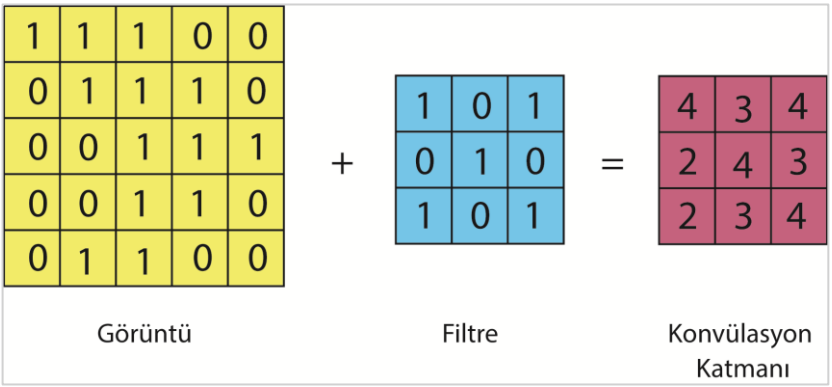
- **Girdi katmanı (Input Layer)**

CNN'deki giriş katmanı, görüntü verilerini içermektedir. Görüntü verileri çok boyutlu matris ile temsil edilir ve bu katman içinde tek bir vektöre yeniden şekillendirilir.

- **Evrişim katmanı (Convolutional Layer)**

Evrişimsel sinir ağlarının temeli konvülasyon (evrişim) katmanıdır. Gerçek değerli bir değişkenin iki fonksiyon üzerindeki işlemidir.

Bu katman dönüşüm katmanı olarak ifade edilebilir. Dönüşüm işlemi, sabit bir filtrenin görüntü üzerinden dolaştırılmasına dayanır. Filtrelerin boyutu, kullanılan görüntü boyutundan küçük olmak zorundadır (Metlek ve Çetiner, 2021). Giriş katmanının boyutlarına göre evrişim işlemlerini gerçekleştirilirken çeşitli filtreler kullanır. Bu filtrelerin boyutu 2×2, 3×3, 5×5 veya 7×7 şeklinde farklı boyutlarda olması muhtemeldir (Uray, 2022). Bu sayede istenilen filtreler görüntülere uygulanmakta ve çıktı görüntü elde edilmektedir. Bu görüntülere özellik haritaları, öznitelik haritaları ya da aktivasyon haritaları denilmektedir (Emek, 2022; Uray, 2022). Şekil3.8'de verilen örnekte 5×5 boyutundaki bir görüntüye 3×3'lük bir filtre uygulandığında sonuç görüntüsü verilmiştir. Bu işlem esnasında kaydırma değeri 1 olacak şekilde matris sonuna kadar devam eder. Son kısma gelindiğinde ise bir satır aşağı kaydırılıp işlem tüm matrisin tamamına uygulana kadar devam eder. Eğer çalışmada RGB üç bantlı bir görüntü kullanıldıysa, verinin yapısından dolayı üç boyutlu bir matris verisi üzerinde dolaştırılarak iki boyutlu bir veri matrisi elde edilmektedir (Metlek ve Çetiner, 2021).



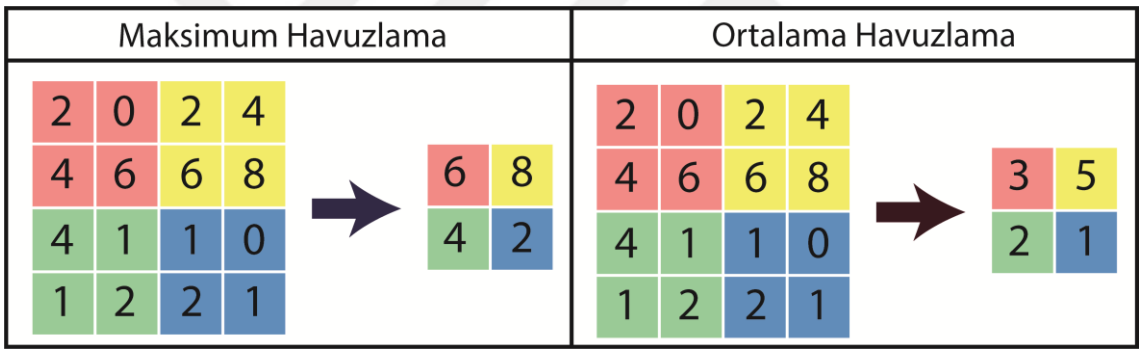
Şekil 3.8. Konvülasyon işlemi ve sonuç görüntüsü

- **Non-Linearity katmanı**

Sisteme doğrusal olmayanlığın (non-linearity) tanıtılması işlemidir.

- **Havuzlama katmanı (Pooling / Downsampling layer)**

Havuzlama katmanı, özellik haritalarının boyutlarını azaltmak için kullanılır (Şennur, 2022). Böylece ağda öğrenilecek parametre sayısını ve yapılan hesaplama miktarını azaltır. Bu durum da işlem hızını arttırmaktadır. Maksimum havuzlama ve ortalama havuzlama en yaygın kullanılanıdır. Maksimum havuzlama işleminde geçerli görünümün maksimum değeri seçilirken, ortalama havuzlamada ortalama değer kullanılır. Şekil 3.9'da iki havuzlamanın temel prensibi 4×4 görüntü üzerinde, 2×2 boyutunda filtre oluşturularak ifade edilmiştir.



Şekil 3.9. Maksimum ve ortak havuzlama işlemi ve sonucu

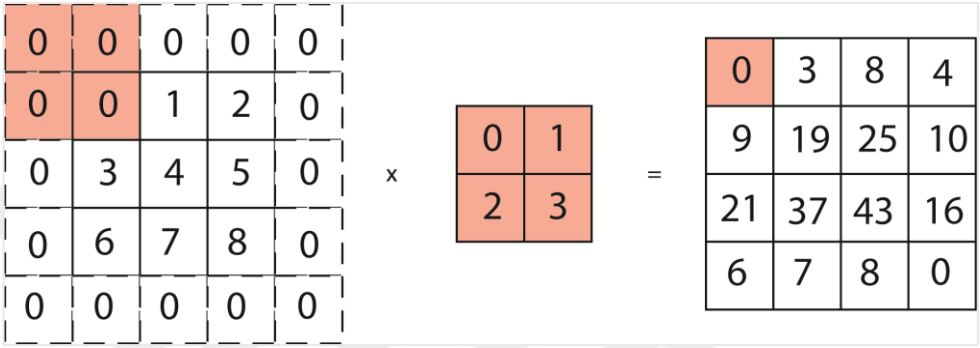
Havuzlama katmanındaki boyutunun küçülmesi bilgi kaybına yol açar (Emek, 2022). Literatürde taranan bazı çalışmalarda, havuzlama yerine adım (stride) tercih edilmiştir. Bu tez çalışmasında maksimum havuzlama tercih edilmiştir. Temel sebebi ise; bir sonraki katmanlarda hesaplama yükünün azalması ve sistemin aşırı öğrenme riskini önlemektir.

- **Adım (Stride)**

Filtrenin ana görsel üzerinde kayma başına kat edilen satır ve sütun sayısı adım olarak ifade edilmektedir. Adım sayısı 1 olmak zorunda değildir.

- **Padding**

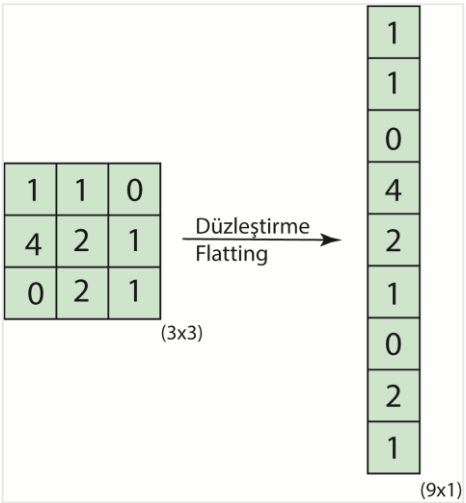
Evrişimli katmanları uygularken problem olan bir konu, görüntünün çevresinde piksel kaybetme eğiliminde olmasıdır. Özellikle birçok ardışık evrişim katmanı uygulanıldığında piksel kayıpları yoğun şekilde yaşanabilir. Bu soruna basit bir çözüm, girdi görüntüsünün sınırının etrafına fazladan dolgu pikselleri eklemektir. Böylece görüntünün etkin boyutunu artırılarak piksel kaybı ortadan kalkacaktır.



Şekil 3.10. Padding işlemi ve sonucu

- **Düzleştirme katmanı (Flattening layer)**

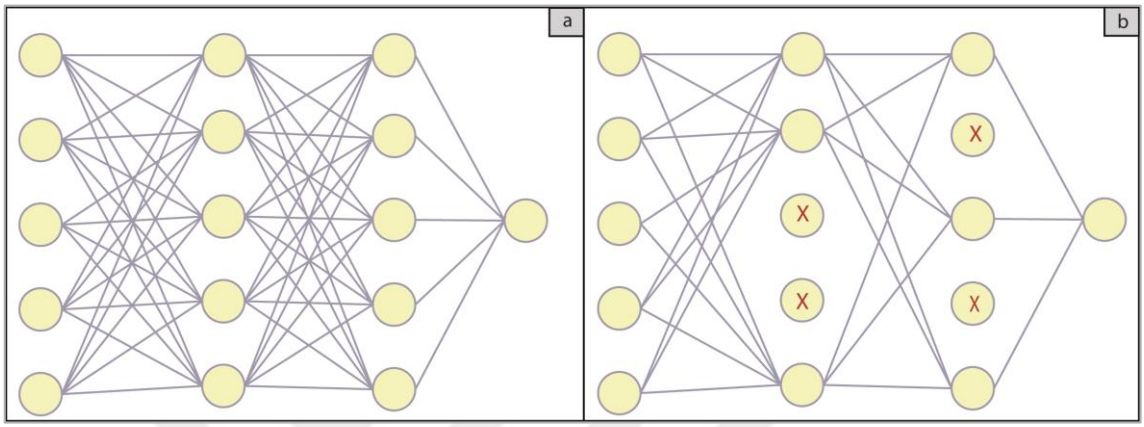
Sinir ağları, giriş verilerini tek boyutlu bir diziden alır. Bu katmanın temel görevi ise, çok boyutlu matrislerin tek boyut diziye dönüştürülmesini sağlamaktır (Şennur, 2022). Tam bağlı katmanın girdi verisini hazırlamak için kullanılan ara ve basit bir katmandır. Düzleştirme işlemine ait görsel Şekil3.11'de sunulmuştur.



Şekil 3.11. Düzleştirme işlemi

- **Dropout katmanı**

ESA'da büyük verilerle eğitim yapıldığında, ağın aşırı öğrenmesi söz konusu olmaktadır. Ağın ezberlenmesini engellemek için dropout katmanı geliştirilmiştir (Hinton ve ark., 2012).



Şekil 3.12. a) Standart basit bir ağ b) Basit bir ağa dropout işleminin uygulanmasından sonra

- **Tam bağlı katman (Fully-Connected layer)**

Evrişimli ağın en son ve en önemli katmanıdır. Sınıflandırma işlemi bu katmanda gerçekleştirilir. Düzleştirme sürecinde almış olduğu girdi verisini, ağlar aracılığıyla besler ve iletir.

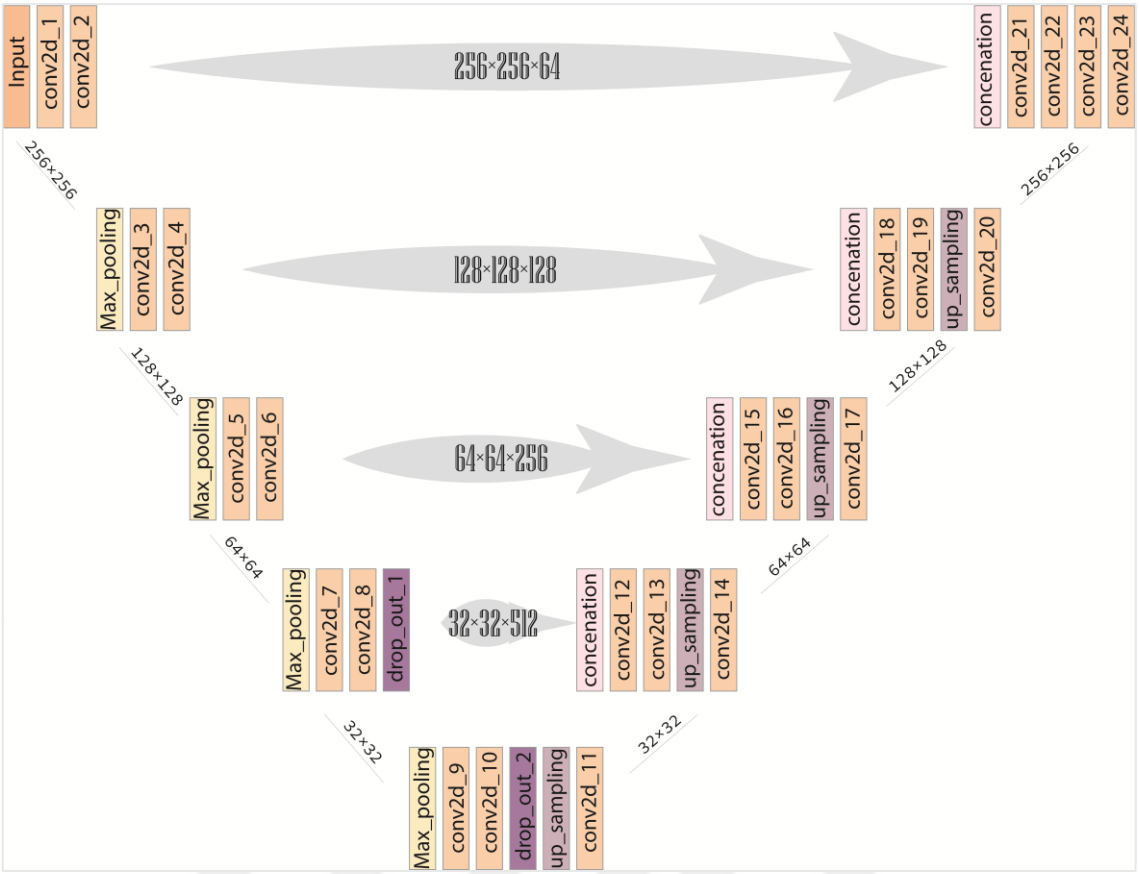
3.3.1.1. U-Net mimarisi

Çalışmada U-Net mimarisi kullanılmıştır. U-Net, tamamen evrişimli ağ temelli bir evrişimli sinir ağıdır ve bu model Ronneberger ve ekibince geliştirilmiştir. Ekip tarafından gerçekleştirilen, "U-Net: Biyomedikal Görüntü Segmentasyonu için Evrişimli Ağlar" başlıklı çalışmalarında, tipik bir sınıflandırma ağıyla karşılaştırıldığında, biyomedikal görüntülerin segmentasyonu için binlerce görüntünün eğitiminde zorluklar olduğunu vurgulamışlardır. Ayrıca, bu tür uygulamalarda genellikle her görüntünün tek bir sınıf etiketine sahip olduğunu belirtmişlerdir. Ancak, biyomedikal alan için bu kadar büyük bir eğitim veri setine ulaşmanın zor olduğunu ve çıktıların lokalizasyon içermesi gerektiğini, yani sınıf etiketlerinin piksellere atanması gerektiğini ifade etmişlerdir (Ronneberger ve ark., 2015). U-Net algoritması, biyomedikal görüntülerdeki semantik

sınıflandırmayı hedefleyerek geliştirilmiş olup, son dönemlerde çok bantlı uydu görüntüleri ve SAR görüntülerinde de etkili bir biçimde kullanılmaktadır. Bu mimari kodlayıcı ve kod çözücü parçalarından oluşan simetrik bir yapıya sahiptir(Uray, 2022). Şekil itibariyle U harfine benzediği için ismini buradan almaktadır(Arıkan ve Yıldız, 2023). Modelin U harfindeki görüntüsü ve yapılan çalışmadaki katman adımları Şekil 3.13'te verilmiştir. Kodlayıcı ve kod çözücü yapıları sayesinde çok ölçekli özellik haritalarını verimli bir şekilde çıkarmakta ve birleştirmektedir(Sariturk ve Seker, 2023). Kodlayıcı kısmında, eğitim verilerinden özelliklerin çıkarılması gerçekleştirilir ve giriş görüntülerinin soyut bir temsilini öğrenmek için bir kodlayıcı blok dizisi kullanılır(Arıkan ve Yıldız, 2023). Giriş katmanına veri girişi doğrudan gerçekleştirilir. İstenen modelin başarısı, başlangıç katmanındaki girdi verisinin boyutuna bağlıdır. Eğer, giriş katmanında görüntü boyutu yüksek olarak verilmişse, daha fazla bellek kullanımı, eğitim süresinin uzaması ve test süresinin her bir görüntü için artması anlamına gelmesi söz konusudur (Emek, 2022). Yapılan çalışmada girdi verisi olarak 256×256 piksel boyutundaki görüntüyü ve aynı boyutlardaki maskelenmiş görüntü kullanılmıştır. Farklı filtreler uygulanması sonucu özellik haritaları oluşturulur. Kodlayıcı kısımda görüntü boyutları küçülürken, kod çözücü kısımda görüntü boyutları artmakta ve en son aşamada girdi verisindeki boyuta denk gelmektedir. Çalışmalarda girdi görüntüsü istenilen boyut seçilebilir.

Bu mimari, özellikle görüntü segmentasyonu, sınıflandırma ve görüntü üzerinde çalışmalara odaklanan gelişmiş bir görüntü segmentasyon algoritması olarak tasarlanmıştır (Yan ve ark., 2022).

Yapay zekâ uygulamalarında, veri setinin aynı anda işlenmesi, modelin doğruluğu ve öğrenme durumu için model eğitiminde optimizasyon parametreleri bulunmaktadır. Bunlar; optimizasyon yöntemleri, aktivasyon fonksiyonları, batch-size ve epoktur. Doğruluk ölçütleri ile model hakkında bilgi edilmesi söz konusudur. İfade edilen seçeneklerden herhangi bir parametrenin değiştirilmesi modelin başarı performansını etkilemektedir (Arıkan ve Yıldız, 2023; Öz, 2021).



Şekil 3.13. Çalışma için oluşturulan U-Net modelinin işlem adımları

3.3.1.1.1 Optimizasyon yöntemleri

Yapay sinir ağlarının eğitimi, matematiksel çözümlerle hata fonksiyonu optimizasyonu problemini ele alarak geliştirilmiştir. Ağdaki ağırlıkların, girdileri doğru çıktı temsillerine hassas bir şekilde dönüştürebilmesi için, eğitim süresince stokastik gradyan azalımı teknikleri kullanılarak sağlıklı düzeltmeler yapılması ve hesaplanabilmesi optimizasyon algoritmalarının geliştirilmesi önemlidir. İyi bir öğrenme için, kayıp fonksiyonunun en küçük değerine ulaşılması önemlidir. Bu en küçük değere ulaşmak için ise optimizasyon algoritmaları kullanılır. Bu algoritmalar, ağın ürettiği çıktı ile gerçek değer arasındaki farkı en aza indirmek için kullanılan yöntemlerdir (Gazel ve Bati, 2019).

3.3.1.1.1.1. SGD (Stochastic gradient descent)

Stokastik" terimi, rastgele bir olasılığa bağlı mekanizmayı veya yöntemi ifade eder; bu sebeple her iterasyonda tüm veri seti yerine birkaç örnek rastgele seçilir(B. Wang ve Ye, 2020). SGD, her eğitim aşamasından sonra ağ yapısını değiştirerek global minimumu bulmayı hedefler. Bu yaklaşım, tüm veri setinin gradyanını hesaplamak yerine rastgele seçilen bir grup için gradyanı yakınsayarak hatayı azaltır. Pratikte, veri setini rastgele karıştırarak ve adım adım küme boyutu ilerleyerek rastgele örnekleme yapılır(Gazel ve Bati, 2019). SGD, sıkça yüksek varyanslı değişiklikler gerçekleştirir, bu da amaç fonksiyonunda önemli varyasyonlara izin verir(Kemal ve Kilicarslan, 2019). Çizelge 3.3'te SGD optimizasyon parametreleri varsayılan değerler, Eşitlik 3.1'de SGD optimizasyonuna ait denklem, verilmiştir (Dewangan ve Sahu, 2021).

Çizelge 3.3. SGD optimizasyon parametresi

Optimizasyon Parametreleri	**Varsayılan Değer**
learning_rate	0.01
momentum	0.0
nesterov	Yanlış
weight_decay	Hiçbiri
clipnorm	Hiçbiri
clipvalue	Hiçbiri
global_clipnorm	Hiçbiri
use_ema	Yanlış
ema_momentum	0.99
ema_overwrite_frequency	Hiçbiri

$$\theta_{t+1} = \theta_t - \alpha \nabla J(Q_t) \tag{3.1}$$

Denklemde, θ_t; t zamanındaki güncellenmek istenen parametrenin değeri, α; öğrenme oranı, $\nabla J(Q_t)$; Kayıp fonksiyonunun θ_t noktasındaki gradyanı, yani parametrelerin ne kadar değiştirilmesi gerektiğini gösteren vektördür.

3.3.1.1.1.2. Adagrad (Adaptive gradient algorithm)

Adagrad, gradyan iniş optimizasyonunu iyileştirmek için Duchi ve arkadaşları tarafından önerilmiştir(Duchi ve ark., 2011). Bu yöntem, öğrenme oranını iterasyonlar boyunca dinamik olarak ayarlayarak adaptif bir yaklaşım sunar. Her bir iterasyonda öğrenme hızını ayarlamak için gradyanların karelerinin toplamına bölerek ilerler. Bu strateji, seyrek veri kümelerinde öğrenme hızının uyum sağlama yeteneğini artırır ve derin öğrenme işlemlerinde etkili sonuçlar elde eder. Ancak, Adagrad'ın bir zayıflığı bulunmaktadır; eğitim ilerledikçe öğrenme hızı zamanla azalır. Bu da yakınsama sürecini etkileyebilir ve doğrulama kaybının belirli dönemlerde yetersiz olmasına yol açabilir. (N. Zhang ve ark., 2018). Kısaca, öğrenme performansının bir süre sonra yetersiz hale gelmesine yol açabilir. Çizelge 3.4'te Adagrad optimizasyon parametreleri varsayılan değerler, Eşitlik 3.2'de Adagrad optimizasyonuna ait güncelleme denklemi, verilmiştir(Dewangan ve Sahu, 2021).

Çizelge 3.4. Adagrad Optimizasyon Parametreleri (URL)

Optimizasyon Parametreleri	**Varsayılan Değer**
learning_rate	0.001
initial_accumulator_value	0.1
epsilon	1e-07
weight_decay	Hiçbiri
clipnorm	Hiçbiri
clipvalue	Hiçbiri
global_clipnorm	Hiçbiri
use_ema	Yanlış
ema_momentum	0.99
ema_overwrite_frequency	Hiçbiri

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\alpha}{\sqrt{G_{t,ii} + \epsilon}} \cdot \nabla J(\theta_{t,i}) \tag{3.2}$$

Denklemde yer alan, $\theta_{t,i}$; t zamanda güncellenmek istenen i parametrenin değeri,
α: öğrenme oranını, $G_{t,ii}$; t zamanda, i parametre bileşeninin geçmiş gradyan karelerinin toplamı, ϵ: sıfıra bölme hatasını önlemek için seçilen küçük bir değer, $\nabla J(\theta_{t,i})$; Kayıp

fonksiyonunun i parametreye göre türevi, yani gradyanı ifade eder. ϵ değeri için, Çizelge3.4'te varsayılan değer olarak 1e-07 ifade edilmiştir.

3.3.1.1.1.3. Adam (Adaptive moment estimation)

Adam optimizasyonu, Kingma tarafından önerilen bir gradyan iniş optimizasyon algoritmasıdır. Bu teknik, gradyanların ilk ve ikinci momentlerinin kestirimlerine dayanarak her bir parametre için farklı uyarlanabilir öğrenme oranlarını hesaplar. Uyarlanabilir moment kestiriminden türetildiği için Adam ismini almıştır (Kingma ve Ba, 2014). Bu metod, RMSprop ve momentumun faydalarını bir araya getirir ve birçok derin öğrenme işleminde etkili sonuçlar verir (N. Zhang ve ark., 2018). Yöntem son zamanlarda popüler olan AdaGrad ve RMSProp yöntemin avantajlarını birleştirmek üzere tasarlanmıştır (Duchi ve ark., 2011; Tieleman ve Hinton, 2018). Adam optimizayonunun bazı avantajları, parametre güncellemelerinin gradyanın ölçeklenmesine bağlı olarak değişmemesi, adımlarının genellikle adım büyüklüğü hiperparametresi ile sınırlı olması, istikrarlı bir hedef gerektirmemesi, seyrek gradyanlarla uyumlu olması ve adım büyüklüğünü doğal bir şekilde azaltma eğilimi göstermesidir(Kingma ve Ba, 2014). Çizelge 3.5'te Adam optimizasyon parametreleri varsayılan değerler, Eşitlik 3.3'te Adam optimizasyonuna ait güncelleme denklemi, verilmiştir(Dewangan ve Sahu, 2021).

Çizelge 3.5. Adam Optimizasyon Parametreleri (URL)

Optimizasyon Parametreleri	**Varsayılan Değer**
learning_rate	0.001
beta_1	0.9
beta_2	0.999
epsilon	1e-07
amsgrad	Yanlış
weight_decay	Hiçbiri
clipnorm	Hiçbiri
clipvalue	Hiçbiri
global_clipnorm	Hiçbiri
use_ema	Yanlış
ema_momentum	0.99
ema_overwrite_frequency	Hiçbiri

$$\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{\widehat{v_t}} + \epsilon} \cdot \widehat{m_t} \tag{3.3}$$

Denklemde, $\widehat{m_t}$; t zamanındaki hareketli ortalama gradyanı, $\widehat{v_t}$: t zamanındaki hareketli ortalama gradyanın karesidir.

3.3.1.1.1.4. Adadelta (Adaptive delta)

Adadelta, Zeiler tarafından önerilen bir gradyan iniş optimizasyon algoritmasıdır (Zeiler, 2012). Ayrıca, azalan öğrenme hızı problemine de çözüm getirebilir ve RMSprop prensibine benzer bir yapıya sahiptir(N. Zhang ve ark., 2018). Bu teknik, manuel bir öğrenme oranı ayarlamasını gerektirmez. Adagrad yöntemindeki dezavantaj olarak görülen hiperparametre seçimine olan duyarlılığın üstesinden gelmek ve öğrenme oranlarının sürekli olarak azalmasını önlemek için geliştirilmiştir(Zeiler, 2012). Çizelge 3.6'da Adadelta optimizasyon parametreleri varsayılan değerler, Eşitlik 3.4'te Adadelta optimizasyonuna ait güncelleme denklemi, verilmiştir(Dewangan ve Sahu, 2021).

Çizelge 3.6. Adadelta Optimizasyon Parametreleri (URL)

Optimizasyon Parametreleri	Varsayılan Değer
learning_rate	0.001
rho	0.95
epsilon	1e-07
weight_decay	Hiçbiri
clipnorm	Hiçbiri
clipvalue	Hiçbiri
global_clipnorm	Hiçbiri
use_ema	Yanlış
ema_momentum	0.99
ema_overwrite_frequency	Hiçbiri

$$\theta_{t+1} = \theta_t - \frac{\sqrt{\widehat{v_t} + \epsilon}}{\sqrt{\widehat{g_t} + \epsilon}} \cdot \widehat{m_t} \tag{3.4}$$

Denlemde verilen, $\widehat{v_t}$; t zamanındaki hareketli ortalama gradyanın karesini, $\widehat{g_t}$; t zamanındaki ikinci moment gradyanı ifade eder.

3.3.1.1.1.5. RMSprop (Root mean square propagation)

Hinton tarafından geliştirilen RMSprop, Adagrad'daki eğitim ilerledikçe öğrenme hızının azalma sorununu ele almak için önerilmiştir(Tieleman ve Hinton, 2018). Bu yöntem, gradyanların karesel ortalamasının üstel bir şekilde azaltılmasıyla öğrenme oranını ayarlar, böylece öğrenme hızının azalmasını önler (Ruder, 2016; Tieleman ve Hinton, 2012; N. Zhang ve ark., 2018). Bazen çalışmalarda momentum dahil edilebilmesi söz konusudur. Adam optimizasyonuna benzemektedir. Fakat momentumlu RMSprop kullanılması durumunda ufak farklılıklar ortaya çıkmaktadır. Momentumlu RMSProp, parametrelerin güncellenmesi için gradyan momentumunu kullanırken, Adam güncellemeleri, gradyanın ilk ve ikinci momentlerinin ortalamasını doğrudan tahmin eder. RMSProp da bias'ların düzeltilmemesi, büyük adım boyutlarına ve sık sık sapmalara yol açabilir (Kingma ve Ba, 2014). Çizelge3.7'de RMSprop optimizasyon parametreleri varsayılan değerler, Eşitlik 3.5'te RMSprop optimizasyonuna ait denklem, verilmiştir(Dewangan ve Sahu, 2021).

Çizelge 3.7. RMSprop optimizasyon parametreleri(URL)

Optimizasyon Parametreleri	**Varsayılan Değer**
learning_rate	0.001
rho	0.9
momentum	0.0
epsilon	1e-07
centered	Yanlış
weight_decay	Hiçbiri
clipnorm	Hiçbiri
clipvalue	Hiçbiri
global_clipnorm	Hiçbiri
use_ema	Yanlış
ema_momentum	0.99
ema_overwrite_frequency	100

$$\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{\hat{v}_t} + \epsilon} \cdot \nabla J(\theta_t) \tag{3.5}$$

3.3.1.1.1.6. Nadam (Nesterov-accelerated adaptive moment estimation)

Adam ve RMSProp için, bu yöntem, optimizasyon stratejilerinin bir birleşimidir(B. Wang ve Ye, 2020). Tasarlanırken, Adam'ın optimizasyon metoduna benzeyen bir yaklaşım kullanılmıştır. Ancak, düz momentum yerine Nesterov'un momentumu tercih edilmiştir (Kemal ve Kilicarslan, 2019; Sutskever ve ark., 2013). Çizelge3.8'de Nadam optimizasyon parametreleri varsayılan değerler, Eşitlik 3.6'da Nadam optimizasyonuna ait denklem, verilmiştir (Dewangan ve Sahu, 2021).

Çizelge 3.8. Nadam optimizasyon parametreleri (URL)

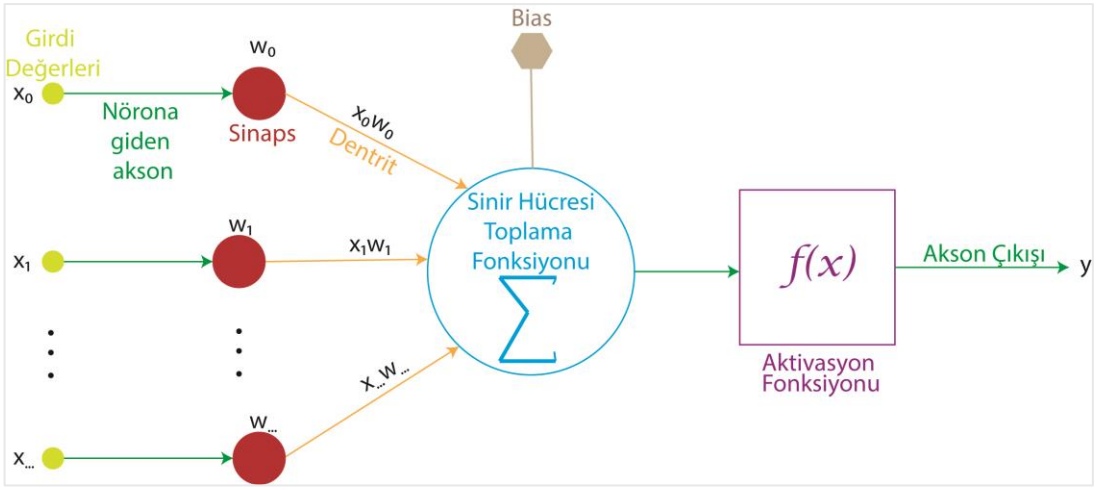
Optimizasyon Parametreleri	Varsayılan Değer
learning_rate	0.001
beta_1	0.9
beta_2	0.999
epsilon	1e-07
weight_decay	Hiçbiri
clipnorm	Hiçbiri
clipvalue	Hiçbiri
global_clipnorm	Hiçbiri
use_ema	Yanlış
ema_momentum	0.99
ema_overwrite_frequency	Hiçbiri

$$\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{\widehat{v_t}} + \epsilon} \cdot (\beta\widehat{m_t} + (1-\beta)\nabla J(\theta_t + \beta\widehat{m_t})) \tag{3.6}$$

Denklemde yer alan, β; Nesterov Momentum'un ağırlıklandırma faktörünü ifade etmektedir. Bu değer Çizelge3.8'de varsayılan değer 0.9 olarak belirlenmiştir.

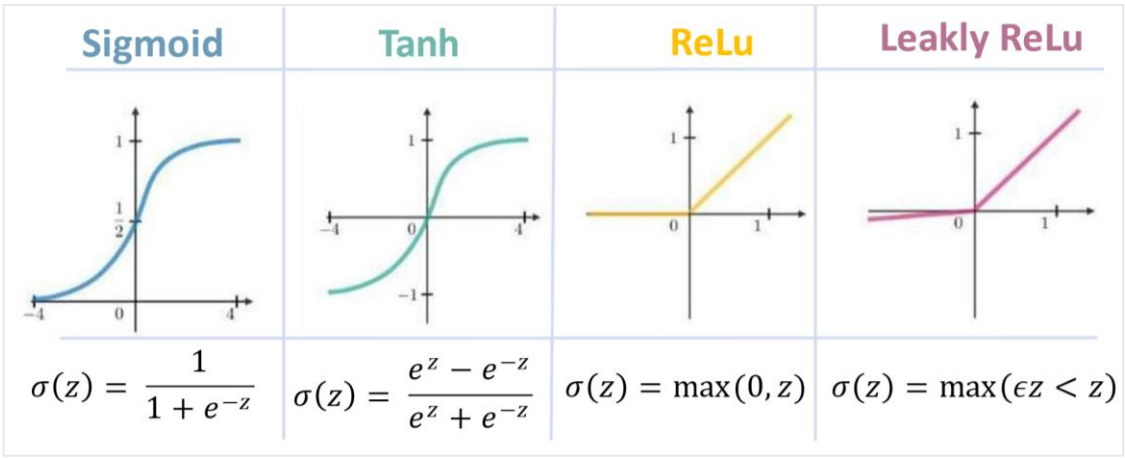
3.3.1.1.2. Aktivasyon fonksiyonları

Yapay bir sinir hücresinde, girdiler (x) ağırlıklarla (w) çarpılır ve bias (b) ile toplanarak hücreye iletilir (Şekil3.14). Bu girdilerin toplamı alınır. Daha sonra, bu toplam değer bir aktivasyon fonksiyonundan geçirilerek düzenlenir(Şennur, 2022).



Şekil 3.14. Aktivasyon fonksiyonun genel yapısı

Aktivasyon fonksiyonları, derin öğrenmede doğrusal olmayan problemleri çözmek için tercih edilirler. Geri beslemeli ağlarda, aktivasyon fonksiyonunun türevi alınır; ancak türevin hesaplanması yoğun işlem gerektirebilir ve bu, ağın genel performansını etkileyebilir. Bu nedenle, türevi kolay hesaplanabilen fonksiyonların seçimi kritiktir. Sigmoid, Hiperbolik Tanjant (tanh), ReLU, Leaky ReLU, gibi yaygın olarak kullanılan aktivasyon fonksiyonlarının grafikleri bulunmaktadır (Şekil 3.15)(Arıkan ve Yıldız, 2023; Uray, 2022). Ayrıca Mish ve Swish gibi farklı aktivasyon fonksiyonları da mevcuttur. Hangi aktivasyon fonksiyonunun kullanılacağı, verinin büyüklüğü, yapısı ve modelin diğer özellikleri gibi faktörler göz önünde bulundurularak belirlenir(Şennur, 2022).



Şekil 3.15. Yaygın kullanılan aktivasyon fonksiyonları ve denklemleri (Arıkan ve Yıldız, 2023; Nwankpa ve ark., 2018)

Literatürde lojistik fonksiyon ya da ezici fonksiyon olarak bilinen sigmoid fonksiyonunda x değeri arttıkça, çıktı değeri de bir sabite yakınsar(Şennur, 2022). Fonsiyonun devamlı olarak türevi alınabiliyorsa, bu fonksiyon girdi verisini öğrenmeye başladığı anlamına gelmektedir(Arıkan ve Yıldız, 2023). Derin öğrenme ağlarında çıkış katmanındaki nöronlarda sıkça tercih edilmektedir. Sigmoid fonksiyonu [0,1] arasında değer almaktadır(Uray, 2022). Dolayısıyla, özellikle karar vermeye yönelik olasılıkların tahmin edilmesinde gereken modeller için oldukça uygundur(Nwankpa ve ark., 2018). Fakat model çok katmanlı bir ağdan oluşuyorsa bu fonksiyonun kullanılmaması tercih edilir. Çalışma kapsamında çıktı fonksiyonu olarak sigmoid tercih edilmiştir.

Sigmoid fonksiyonuna yapı olarak benzeyen bir diğer fonksiyon ise tanh'tır. Bu fonksiyonda [-1,1] aralığında değer almakta ve orijin etrafında simetrik bir yapıya sahiptir(Nwankpa ve ark., 2018). Tanh fonksiyonunun en önemli problemi, uç noktalarda türevi sıfıra yaklaştığı için gradyan kaybı sorunuyla karşılaşır. Bu nedenle, çok katmanlı ağlarda genellikle tercih edilmez (Şennur, 2022).

Sigmoid ve tanh fonksiyonlarına göre daha hızlı verimli çalışan ReLu fonksiyonu sinir ağlarında ara katmanlarda kullanılmaktadır(Ramachandran ve ark., 2017; Zeiler ve ark., 2013). ReLU fonksiyonunun değer aralığı ise [0,+∞)'dur (Uray, 2022). ReLu fonksiyonun pozitif olmasında, geri yayılım sırasında etkin bir işlem gerçekleşir, ancak negatifse, sadece sıfır olmasının dezavantajı vardır(Madhu ve ark., 2023). Bu da, yayılım sürecinde öğrenmenin gerçekleşmeyeceği anlamına gelir(Şennur, 2022). Bu dezavantaj durumu Leakly ReLu fonksiyonunda aşılmıştır.

Leaky ReLu, ReLu fonksiyonunun kısıtlamasını hafifletmek için tasarlanmıştır; bu sayede negatif girdiler için sıfır olmayan bir gradyanı vardır(Maas ve ark., 2013). Leaky ReLU'da negatif değerler neredeyse sıfıra yakın olsa da, tam olarak sıfır değildir (Şennur, 2022). Çünkü, negatif bölgedeki değerleri 0.01 katsayısı ile çarpmakta ve öğrenme işlemi bu şekilde gerçekleşmiş olmaktadır(Uray, 2022).

3.3.1.1.3.Küme boyutu (Batch size)

Model eğitilirken veriler belirli sayıda gruplara bölünür ve eğitime bu gruplar dahil edilir. Küme boyutu, modelin aynı anda işleyeceği veri miktarını belirleyen bir hiper parametredir (Şennur, 2022). Bir batch, bir modelin güncellenmesi için kullanılan veri örneklerinin aynı anda işlenen bir kümesidir. Eğitim veri seti genellikle çok büyük

olduğundan, tüm veri setini tek seferde işlemek zor olabilir. Bu nedenle, veri seti küçük gruplara bölünür ve bu küçük gruplar " batch " olarak adlandırılır.

Batch büyüklüğü, eğitim sırasında belirlenen bir hiperparametredir ve genellikle deneme-yanılma yöntemiyle belirlenir. Küçük batch'ler daha az bellek kullanır ve daha hızlı eğitim sağlayabilir, ancak daha fazla gürültülü gradyanlara ve daha az stabiliteye yol açabilir. Büyük batch'ler daha stabil gradyanlar sağlayabilir, ancak daha fazla bellek kullanır ve daha yavaş eğitim sürelerine neden olabilir. Optimal parti büyüklüğü, modelin karmaşıklığına, kullanılan donanıma ve eğitim veri setinin boyutuna bağlı olarak değişebilir(Arıkan ve Yıldız, 2023).

3.3.1.1.4. Epok

Modelde, öncelikle bir küme eğitilir, ardından modelin performansı değerlendirilir ve geriye yayılım kullanılarak ağırlıklar güncellenir. Bu işlemler tekrarlanarak en uygun ağırlık değerleri belirlenmeye çalışılır. Her bir bu eğitim döngüsüne 'epok' adı verilir (Şennur, 2022). Bir epok, tam veri setinin model tarafından bir kez geçirilmesini ifade eder. Derin öğrenme modelleri genellikle büyük veri setleri üzerinde eğitildiği için, veri seti üzerinde bir kez geçmek modelin tüm verileri görmesini sağlamaya yeterli olmayabilir. Bu nedenle, eğitim süreci genellikle birden fazla epok boyunca gerçekleştirilir. Her epok, modelin ağırlıklarının güncellenmesi ve eğitimin ilerlemesi için bir fırsat sağlamaktadır.

Epok sayısı, eğitim sırasında bir hiperparametre olarak belirlenir ve genellikle deneysel yollarla belirlenir. Modelin ne kadar karmaşık olduğuna, veri setinin boyutuna ve eğitim hedeflerine bağlı olarak uygun epok sayısı değişebilir. Epok sayısının artırılması, modelin daha fazla eğitilmesini sağlayabilir ancak aşırı uyum (overfitting) riskini artırabilir. Bu nedenle, eğitim sürecini optimize etmek için epok sayısı dikkatlice seçilmelidir, genellikle doğrulama veri seti üzerindeki performans metrikleri kullanılarak belirlenir.

Batch size ve epok sayısı genellikle birbirine karıştırılan ancak farklı anlamlar taşıyan iki terimdir. Batch size, öğrenme işlemi sırasında her bir iterasyon için hesaplamaların yapıldığı veri kümesinin parçalara bölünmüş boyutunu ifade ederken, epok sayısı ise bir öğrenme sürecinin tamamlanma sayısını temsil eder. Batch size, birden fazla girdi verisinin gruplar halinde işlenmesini sağlar. Bu şekilde, her iterasyon sırasında ağırlık değerleri güncellenir ve hata oranları minimize edilir. Basitçe ifade etmek

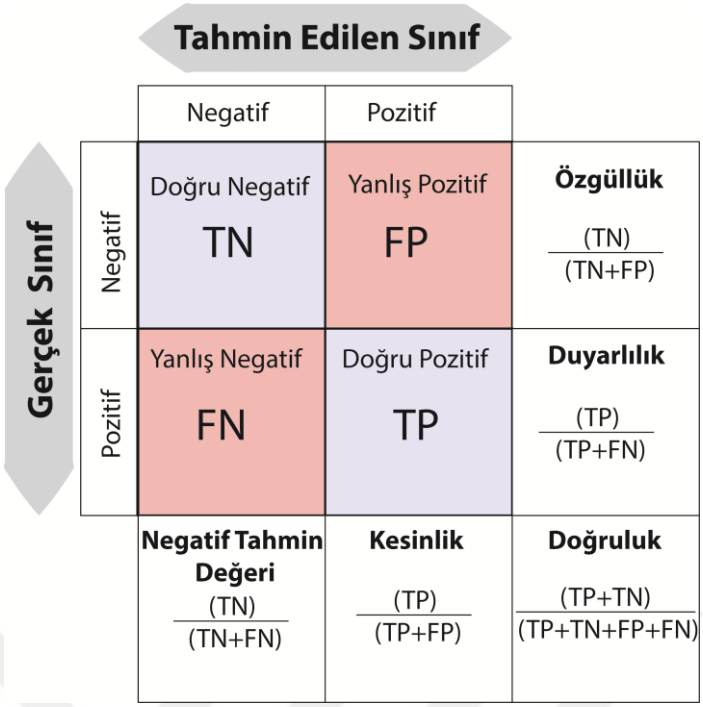
gerekirse, batch size verilerin her bir adımda işlenme miktarını belirtirken, epok sayısı tam bir eğitim veri setinin kaç kez işleneceğini belirtir. Örneğin; 2500'ün eğitim veri setinin batch size değeri 5 olarak seçildiğinde, bu veri seti 500'lük gruplara bölünür ve 1 epoğu tamamlamak için 5 kez batch size işlemi gerçekleştirilir (Arıkan ve Yıldız, 2023).

3.3.1.1.5. Öğrenme oranı

Öğrenme oranı, ağdaki ayarların ne sıklıkta güncelleneceğini belirler. Eğitim sırasında, ağdaki ağırlıkların güncellenmesi, bir hata fonksiyonundan elde edilen hata sinyali ve öğrenme oranı kullanılarak gerçekleştirilir (Bayramoğlu, 2021). Kullanıcılar genellikle deneme-yanılma yöntemiyle bu hızı belirlerler. Bu değer genellikle 0.1 ile 0.000001 arasında seçilir, ancak sabit olabileceği gibi artan veya azalan şekilde de seçilebilir (Saralioğlu, 2020, Karademir, 2019). Büyük bir öğrenme oranı, ağırlıkların daha büyük adımlarla güncelleneceği anlamına gelir, bu da daha hızlı öğrenme sağlayabilir ancak istikrarsızlık riskini artırabilir. Küçük bir öğrenme oranı ise daha küçük adımlarla güncelleme yapar, bu da daha istikrarlı ancak daha yavaş bir öğrenme sürecine neden olabilir. Ayrıca aşırı öğrenme riski taşımaktadır (Yılmaz, 2020; Yılmaz ve Kavzoğlu, 2021). Bu nedenle öğrenme oranı, modelin doğruluğunu ve genelleme yeteneğini etkileyen kritik bir faktör olduğu anlaşılmaktadır. Uygun bir öğrenme oranı seçmek, modelin eğitimini optimize etmek için önem arz etmektedir.

3.3.1.1.6. Doğruluk ölçütleri (Metrikleri)

Sınıflandırma performansının ölçülmesi için çeşitli metrikler bulunmaktadır. Bu metrikler, gerçek değerler ile yapılan tahminlerin karşılaştırılmasıyla hesaplanır. Sonuçlar genellikle bir hata matrisi şeklinde sunulur. Hata matrisinde satırlar gerçek değerleri, yani beklenen değerleri temsil ederken, sütunlar tahmin edilen değerleri gösterir (Kırcalı, 2022; Uray, 2022). Şekil 3.16'da verilen hata matrisi; Doğru Negatif (TN), Yanlış Pozitif (FP), Yanlış Negatif (FN) ve Doğru Pozitif (TP) değerlerinden oluşmaktadır.



Şekil 3.16. Doğruluk metriklerinin gösterimi

3.3.1.1.6.1 Genel doğruluk (Overall accuracy – OA)

Doğru sınıflandırılmış piksellerin, toplam piksellere oranını ifade eden ölçüdür. Eşitlik 3.7'de doğruluk hesabı matematiksel olarak ifade edilmiştir.

$$Doğruluk = \frac{(TP + TN)}{(TN + FP + FN + TP)} \tag{3.7}$$

3.3.1.1.6.2 Duyarlılık (Recall)

Doğru olarak tahmin edilmesi gereken piksellerin ne kadarını doğru sınıfa atadığını gösteren metriktir. Eşitlik 3.8'de duyarlılık hesabı matematiksel olarak ifade edilmiştir.

$$Duyarlılık = \frac{TP}{(TP + FN)} \tag{3.8}$$

3.3.1.1.6.3 Kesinlik (Precision)

Tahmin edilen piksellerin ne kadarının doğru sınıfa ait olduğunu gösterir. Eşitlik 3.9'da kesinlik hesabı matematiksek olarak ifade edilmiştir.

$$Kesinlik = \frac{TN}{(TN + FP)} \tag{3.9}$$

3.3.1.1.6.4 F1-skoru (F1-score)

Duyarlılık ve kesinlik metriklerinin harmonik ortalamasının alınması ile hesaplanmaktadır. Eşitlik 3.10'da kesinlik hesabı matematiksek olarak ifade edilmiştir.

$$F_1\ Skor = \frac{2 \times Duyarlılık \times Kesinlik}{Duyarlılık + Kesinlik} \tag{3.10}$$

3.3.1.1.6.5 Jaccard skoru (Intersection over union - IoU)

Jaccard katsayı benzerlik ölçmek amacıyla kullanılan bir istatiksel yöntemdir (Jun Zhang ve ark., 2019). İki kümenin kesişimlerinin, birleşimlerine oranıyla hesaplanmaktadır. Doğruluk değeri [0,1] aralığındadır. Matematiksel formülü Eşitlik 3.11'de verilmiştir.

$$IoU = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{3.11}$$

Eşitlik 3.11'de kullanılan A gerçek maskelenmiş görüntüyü, B ise model sonucunda elde edilen tahmin görüntüsüdür.

3.3.1.1.6.6 Sørensen-Dice benzerlik katsayı indeksi

Dice-Benzerlik katsayısı gerçek maskelenmiş görüntü ile tahmin edilen görüntü arasındaki çakışma derecesini ifade etmektedir. Jaccard skorunda olduğu gibi [0,1] aralığında değer almaktadır. Matematiksel olarak Eşitlik 3.12'de ifade edilmiştir. Çıkan

katsayı sonucunun 1'e yakın olması model sınıflandırmasının yüksek doğrulukta olduğunu ifade ederken, 0'a yakın olması da düşük doğrulukta olduğunu ifade etmektedir(Jun Zhang ve ark., 2019).

$$DBK = \frac{2 \times |A \cap B|}{|A| + |B|} \tag{3.12}$$

Eşitlikte kullanılan A gerçek maskelenmiş görüntüyü, B ise model sonucunda elde edilen tahmin görüntüsüdür.

3.3. Çalışma İşlem Adımları

Çalışmada Ankara ilinin bazı kısımlarındaki yol ve bina ağlarını U-Net mimarisi kullanılarak otomatik çıkarma işlemi gerçekleştirilmiştir (Şekil 3.16). Tez kapsamında bu mimarinin seçilmesinin temel nedeni: literatürdeki bir çok görüntü sınıflandırma çalışmasında kullanılmış olması ve mimarinin bu amaç doğrultusunda geliştirilmiş olmasıdır (Ronneberger ve ark., 2015). Çalışma kapsamında U-net modeli Python v3.5 programlama dili kullanılarak geliştirilmiştir. Python, ücretsiz ve açık kaynaklı programlama dilidir. Yaygın olarak kullanıldığı için geniş bir kullanıcı kitlesine sahiptir. Problem karşısında bilgi ve donanıma kolay bir şekilde erişim imkanı bulunmaktadır (Kırcalı, 2022). Yazılım, Tensorflow ve Keras gibi derin öğrenme kütüphanelerinden yararlanılarak grafik işlemciyi kullanacak şekilde tasarlanmıştır. Python kütüphanelerinin yaygın olarak kullanılanları ve nasıl kullanıldıkları, çalışma sürecinin bir parçası olarak ele alınmış ve Çizelge 3.9'da sunulmuştur. Grafik işlemcinin tercih edilmesinin nedeni, özellikle görüntü işleme problemlerinde normal işlemcilere kıyasla önemli ölçüde daha hızlı hesaplama yapabilmesidir. Uygulama için Intel(R) Core (TM) i7-10750, 32,0 GB bellek ve NVIDIA GTX 850M GPU MSI dizüstü bilgisayar kullanıldı. Çalışma veri hazırlığı, modelin oluşturulması ve eğitilmesi, görüntülerin tahmin edilmesi, sonuç değerlerin üretilmesi ve değerlendirme aşamasından oluşmaktadır.

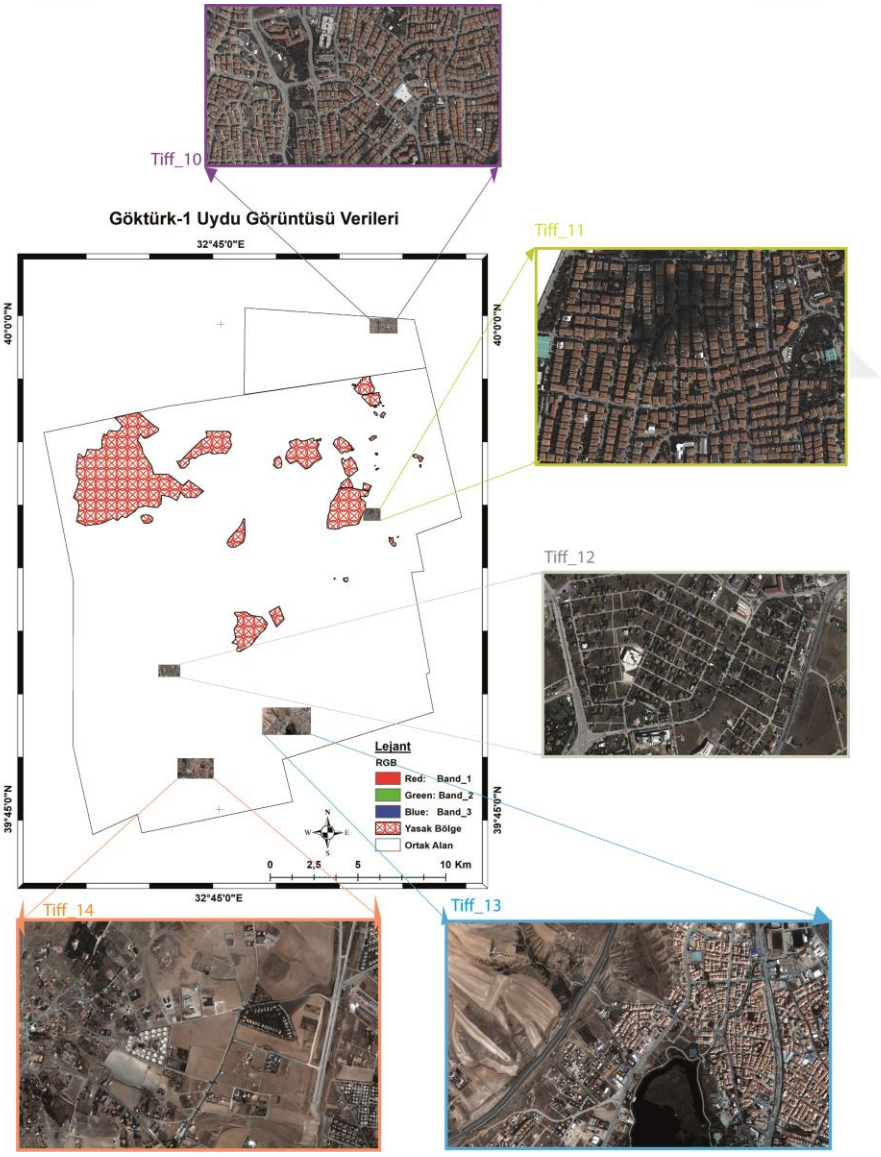
Çizelge 3.9. Kullanılan kütüphaneler ve amaçları

Kütüphane İsmi	Kullanım Amacı	Alıntı
Numpy	Çok boyutlu dizilerin (arrays) oluşturulması, vektörel ve matris işlemlerini hızlı bir şekilde gerçekleştirilmesi, büyük veri kümeleri üzerinde etkili hesaplamalar yapmak ve matris	(Numpy, 2024)

	çarpımı, determinant hesaplama, ters matris bulma için idealdir.	
Matplotlib	Grafikler, çizimler ve görselleştirmeler oluşturmak için kullanılan bir kütüphanedir. Oluşturulan görselleştirmeleri farklı dosya biçimlerine (PNG, JPEG, PDF, SVG, vb.) kaydetmek veya farklı arayüzlerde görüntülemek için esnek bir yapı sunar.	(Matplotlib, 2024)
Keras	Derin öğrenme modelleri oluşturmak, eğitmek ve değerlendirmek için kullanılan bir yüksek seviye derin öğrenme kütüphanesidir. Bu kütüphane sayesinde kullanıcılar farklı katmanları, aktivasyon fonksiyonlarını, optimizasyon algoritmalarını ve diğer bileşenleri kolayca bir araya getirebilmektedirler.	(Keras, 2024)
Tensorflow	Google tarafından geliştirilen açık kaynaklı bir makine öğrenimi ve derin öğrenme kütüphanesidir. Yapay sinir ağları, evrişimli sinir ağları, yinelemeli sinir ağları ve diğer derin öğrenme modellerini oluşturmak ve eğitmek için kullanılmaktadır. Ayrıca, modelleri görselleştirmek, eğitim sürecini izlemek ve sonuçları analiz etmek için TensorBoard adlı bir araçla entegre şekilde çalışma imkânı da bulunmaktadır.	(Tensorflow, 2024)
OpenCV	Çeşitli görüntü formatlarını okumak ve yazmak, görüntüler üzerinde parlaklık, kontrast ayarlama, renk dönüşümleri, bulanıklık giderme, kenar tespiti, şekil tanıma gibi işlemlerde kullanılmaktadır. Görüntü analizi ve özellik çıkarımı için çeşitli algoritmalar içermekte ve görüntü segmentasyonu, kenar tespiti, köşe tespiti gibi işlemlerin gerçekleştirilmesine imkân sağlamaktadır. Görüntülere ek olarak video izleme, hareketli nesneleri izleme ve güvenlik uygulamaları için de tercih edilmektedir.	(OpenCV, 2024)
os	Dosya oluşturma, klasör oluşturma, dosya veya klasör silme, dosya adı değiştirme gibi işlemleri gerçekleştirilmesi bu kütüphane sayesindedir.	(os, 2024)
h5py	Büyük boyutlu verilerin depolanması, paylaşılması ve işlenmesi için kullanılan bir dosya formatıdır ve özellikle bilimsel hesaplama ve veri analizi alanlarında yaygın olarak kullanılmaktadır. HDF5 dosyaları, gruplar ve veri kümeleri olarak organize etmeye olanak sağlamaktadır.	(h5py, 2024)

Çalışma bölgesi ve veriler hakkında bilgi verilmişti. Belirtilen çalışma bölgesi 5 kısma ayrılmıştır. İnceleme kapsamında Türkiye'nin başkenti olan şehirde, yoğun yapılaşmanın gözlendiği kentsel bölgeler ve bazı bölgelerinde gecekondu mahallelerinin bulunduğu alanlar detaylı bir şekilde ele alınmıştır.

Bina kavramı, yapılar ve evler de dahil olmak üzere düzenli yapay özellikler olan tüm inşa edilmiş nesneleri ifade eder. Çoğu bina esas olarak dikdörtgenlerden veya türevlerinden oluşur (Zheng ve ark., 2020). Yol ise, binalar arasında bağlantı sağlayan, taşıma ve ulaşım için kullanılan bir altyapıdır. Genellikle, şehirlerde, kasabalarda ve kırsal bölgelerde yer alan binalar arasında ulaşımı kolaylaştırmak için planlanan ve inşa edilen yollar, toplu taşıma ağlarına, araç trafiğine veya yaya trafiğine hizmet edebilir.



Şekil 3.17. Çalışma bölgesine beş görüntü

Göktürk-1 uydu görüntüsü RGB ve NIR bandı olmak üzere 4 bant bulunmaktadır. Çalışmada bant sayısını 3'e indirmek için Global Mapper programından yararlanıldı. Ve tüm görüntüler bant sayısı 3 (RGB) olacak şekilde ayarlandı. Hesaplama maliyetini azaltmak ve veri kümesinin boyutunu artırmak için görüntüler ve maskeler 256 × 256 piksel boyutuna indirgendi. Kullanılan ağ 39 katmanlı (1 giriş katmanı, 24 evrişim katmanı, 4 maksimum havuzlama, 4 üst örnekleme, 1 dropout, 4 birleştirme) evrişimli bir sinir ağıdır. Eğitilebilen toplam parametre sayısı ise, 31 032 837'dir. Eğitime ve katmanlara ait ayrıntılı bilgiler Şekil 3.17'de verilmiştir.

Model: "model_UNet"

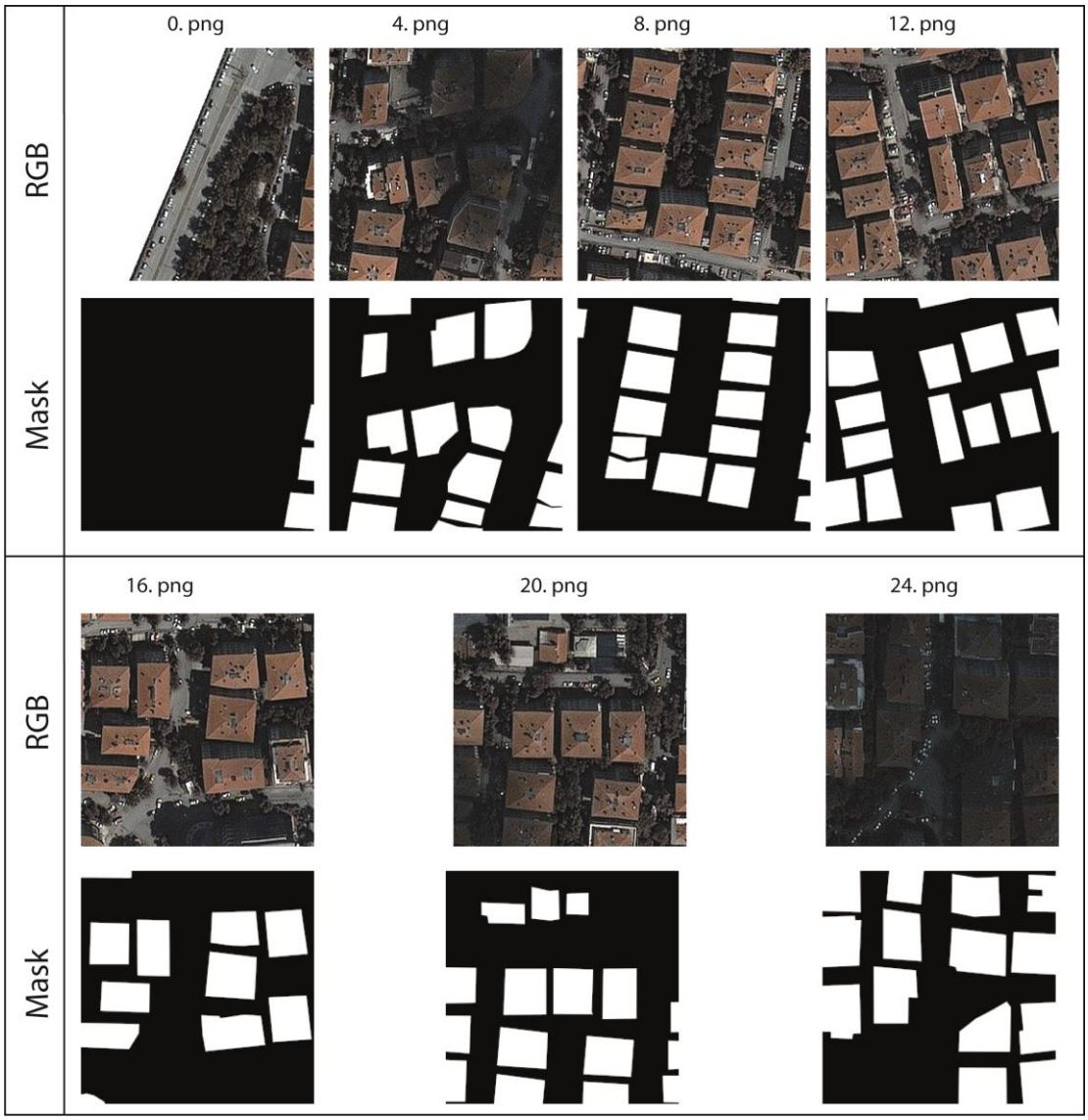
Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 256, 256, 3)]	0	[]
conv2d (Conv2D)	(None, 256, 256, 64)	1792	['input_1[0][0]']
conv2d_1 (Conv2D)	(None, 256, 256, 64)	36928	['conv2d[0][0]']
max_pooling2d (MaxPooling2D)	(None, 128, 128, 64)	0	['conv2d_1[0][0]']
conv2d_2 (Conv2D)	(None, 128, 128, 128)	73856	['max_pooling2d[0][0]']
conv2d_3 (Conv2D)	(None, 128, 128, 128)	147584	['conv2d_2[0][0]']
max_pooling2d_1 (MaxPooling2D)	(None, 64, 64, 128)	0	['conv2d_3[0][0]']
conv2d_4 (Conv2D)	(None, 64, 64, 256)	295168	['max_pooling2d_1[0][0]']
conv2d_5 (Conv2D)	(None, 64, 64, 256)	590080	['conv2d_4[0][0]']
max_pooling2d_2 (MaxPooling2D)	(None, 32, 32, 256)	0	['conv2d_5[0][0]']
conv2d_6 (Conv2D)	(None, 32, 32, 512)	1180160	['max_pooling2d_2[0][0]']
conv2d_7 (Conv2D)	(None, 32, 32, 512)	2359808	['conv2d_6[0][0]']
dropout (Dropout)	(None, 32, 32, 512)	0	['conv2d_7[0][0]']
max_pooling2d_3 (MaxPooling2D)	(None, 16, 16, 512)	0	['dropout[0][0]']
conv2d_8 (Conv2D)	(None, 16, 16, 1024)	4719616	['max_pooling2d_3[0][0]']
conv2d_9 (Conv2D)	(None, 16, 16, 1024)	9438208	['conv2d_8[0][0]']
dropout_1 (Dropout)	(None, 16, 16, 1024)	0	['conv2d_9[0][0]']
up_sampling2d (UpSampling2D)	(None, 32, 32, 1024)	0	['dropout_1[0][0]']
conv2d_10 (Conv2D)	(None, 32, 32, 512)	2097664	['up_sampling2d[0][0]']
concatenate (Concatenate)	(None, 32, 32, 1024)	0	['dropout[0][0]','conv2d_10[0][0]']
conv2d_11 (Conv2D)	(None, 32, 32, 512)	4719104	['concatenate[0][0]']
conv2d_12 (Conv2D)	(None, 32, 32, 512)	2359808	['conv2d_11[0][0]']
up_sampling2d_1 (UpSampling2D)	(None, 64, 64, 512)	0	['conv2d_12[0][0]']
conv2d_13 (Conv2D)	(None, 64, 64, 256)	524544	['up_sampling2d_1[0][0]']
concatenate_1 (Concatenate)	(None, 64, 64, 512)	0	['conv2d_5[0][0]','conv2d_13[0][0]']
conv2d_14 (Conv2D)	(None, 64, 64, 256)	1179904	['concatenate_1[0][0]']
conv2d_15 (Conv2D)	(None, 64, 64, 256)	590080	['conv2d_14[0][0]']
up_sampling2d_2 (UpSampling2D)	(None, 128, 128, 256)	0	['conv2d_15[0][0]']
conv2d_16 (Conv2D)	(None, 128, 128, 128)	131200	['up_sampling2d_2[0][0]']
concatenate_2 (Concatenate)	(None, 128, 128, 256)	0	['conv2d_3[0][0]','conv2d_16[0][0]']
conv2d_17 (Conv2D)	(None, 128, 128, 128)	295040	['concatenate_2[0][0]']
conv2d_18 (Conv2D)	(None, 128, 128, 128)	147584	['conv2d_17[0][0]']
up_sampling2d_3 (UpSampling2D)	(None, 256, 256, 128)	0	['conv2d_18[0][0]']
conv2d_19 (Conv2D)	(None, 256, 256, 64)	32832	['up_sampling2d_3[0][0]']
concatenate_3 (Concatenate)	(None, 256, 256, 128)	0	['conv2d_1[0][0]','conv2d_19[0][0]']
conv2d_20 (Conv2D)	(None, 256, 256, 64)	73792	['concatenate_3[0][0]']
conv2d_21 (Conv2D)	(None, 256, 256, 64)	36928	['conv2d_20[0][0]']
conv2d_22 (Conv2D)	(None, 256, 256, 2)	1154	['conv2d_21[0][0]']
conv2d_23 (Conv2D)	(None, 256, 256, 1)	3	['conv2d_22[0][0]']

Total params: 31,032,837
Trainable params: 31,032,837
Non-trainable params: 0

Şekil 3.18. U-net mimarisinin işlem adımları

Model de evrişim katmalarında ReLu aktivasyon fonksiyonu kullanılırken, en son çıkış katmanında sigmoid fonksiyonu kullanılmıştır.

4 bölge için oluşturulan görüntülerden birkaç tanesi seçilmiş ve işlem sonuçları bu görüntülerden üzerinden ifade edilecektir. Şekil 3.18'de Tiff_11, RGB ve maskelenmiş görüntüler verilmiştir. Tiff_10, Tiff_12 ve Tiff_13'ün RGB ve maskelenmiş görüntüleri EK'te sunulmuştur. Görüntülerin 256 boyutlarına indirilmesinde uzantı durumu kontrol edilmelidir. Geotiff ya da tiff olması image size komutu ile bölünmesinde önem arz etmektedir.



Şekil 3.19. Göktürk-1 Tiff_11 RGB ve Mask görüntüleri

4. ARAŞTIRMA SONUÇLARI VE TARTIŞMA

Bu bölümde, bina ve yol çıkarım görevlerinde farklı hiper parametrelerin etkilerini değerlendiren çalışmalar sunulmuştur. İlk kısımda, 3 farklı batch boyutunun modelin eğitim süresi ve doğruluğuna olan etkileri detaylı bir şekilde incelenmiştir. Sonraki kısımda, öğrenme oranındaki değişikliklerin modelin hızı ve genel doğruluğu üzerindeki etkileri tartışılmıştır. Üçüncü kısımda, bina tespiti için farklı optimizasyon algoritmalarının performansları karşılaştırılmıştır. Bina için altı adet optimizasyon, yol ağları için de iki optimizasyon değerlendirilmiştir. Yol ağlarında iki optimizasyon seçilmesinin nedeni, bina çıkarımda Adam ve RMSprop sonuçlarının diğerlerine yüksek gelmesinden dolayıdır. Çeşitli optimizasyonları kullanarak her bir yöntemin model başarımına katkısı ve etkinliği incelenmiştir. Dördüncü bölümde ise, farklı eğitim sayılarının model performansı üzerindeki etkileri analiz edilmiştir. Eğitim sayısının arttırılmasının doğruluk ve model genellemesi üzerindeki etkileri ele alınmıştır. Benzer işlemler yol ağları içinde gerçekleştirilmiştir.

4.1. Bina için Farklı Batch-Boyutu'nun Sonuç Değerlendirmesi

Batch boyutu durumunu incelemek için, 3 farklı durum ele alınmıştır. Bunlar; 8,16 ve 32'dir. Bu optimizasyondaki farklılığı anlamak amacıyla, Adam optimizasyonu ve öğrenme oranı 1e-3 sabit olacak olacak şekilde seçilmiştir. Şekil4.1'de Batch Size 8, Şekil4.2'de Batch Size 16 ve Şekil4.3'te Batch Size 32'de eğitim modelinin loss, dice ve Iou grafikleri ifade edilmiştir. Grafikler 4 farklı eğitim süresine göre sunulmuş: 100 epok, 50 epok, 25 epok ve 10 epok. Her eğitim süresi için 2 grafik çiftiyle ifade edilmektedir. Loss grafiği, modelin eğitim verileri üzerindeki kaybı ifade ederken, Val_loss modelin doğrulama verileri üzerindeki kaybını göstermektedir. Doğrulama kaybı, modelin genellenebilirliğini değerlendirmek için kullanılır. Eğitim sayısı arttıkça kaybın azalması beklenmektedir. Dice Coef grafiği, modelin bina segmentasyonu için Dice katsayısını gösterir. Dice katsayısı, tahmin edilen ve gerçek maskeler arasındaki örtüşmeyi ölçer. 1'e yakın değerler daha iyi performansı gösterir. Val Dice Coef de modelin bina segmentasyonu için doğrulama verileri üzerindeki Dice katsayısını gösterir. IoU Coef ise Dice Coef'e benzemektedir. Bu grafik, modelin bina segmentasyonu için kesişim (IoU) katsayısını gösterir. IoU katsayısı, tahmin edilen ve gerçek pikseller arasındaki örtüşmeyi ölçer. 1'e yakın değerler daha iyi performansı gösterir. Val IoU Coef, modelin bina

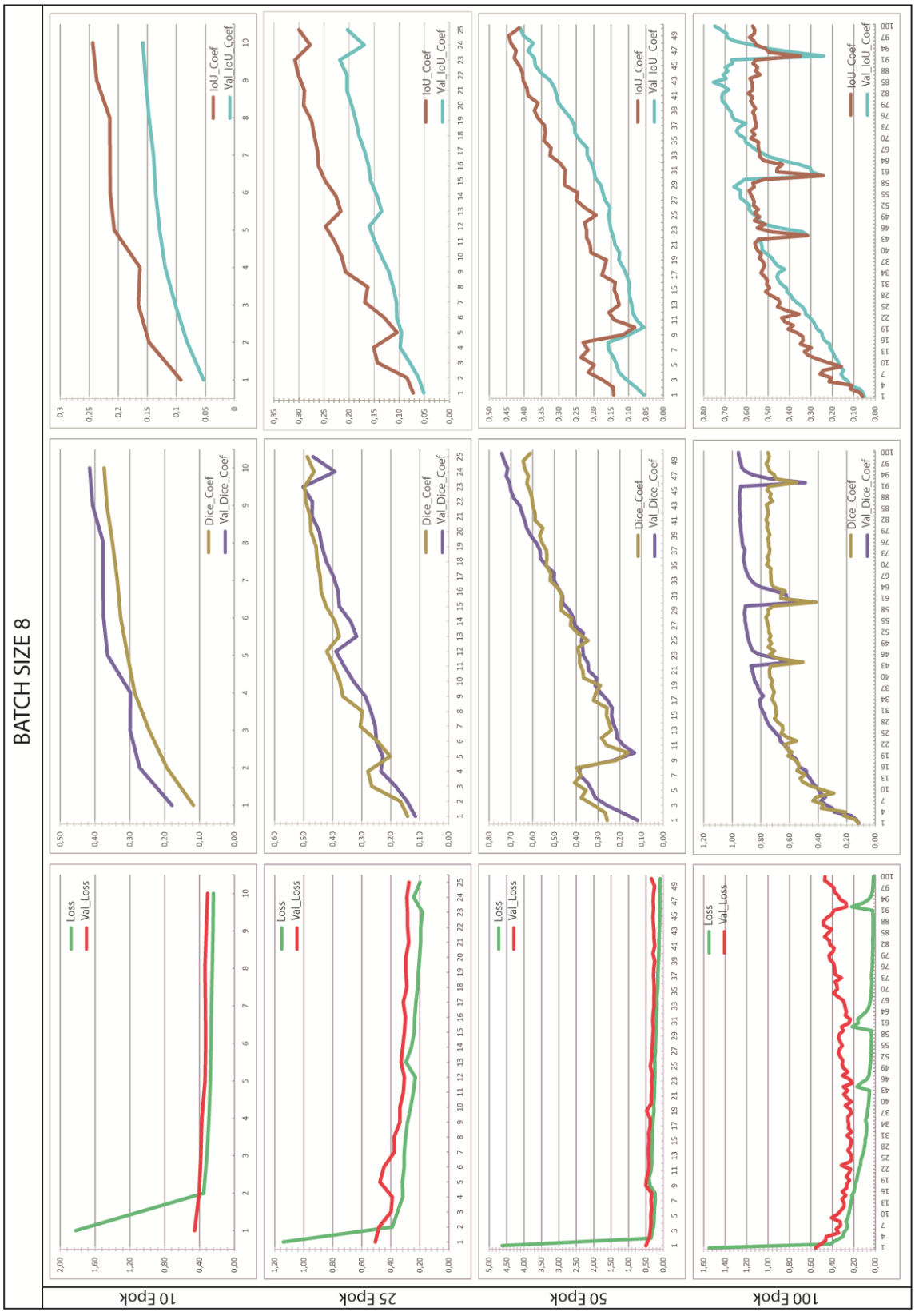
segmentasyonu için doğrulama verileri üzerindeki IoU katsayısını gösterir. Eğitim sayısı arttıkça Dice Coef ve IoU Coef değerlerinin artması beklenmektedir.

Batch boyutu 8 optimizasyonu için en düşük doğruluk 10 epokta iken, en yüksek doğruluk 100 epokta gözükmektedir. Kayıp fonksiyonu 10 epok sonunda 0,2400; 25 epokta 0,1991; 50epokta 0,0893 ve 100 epokta 0,0172 seviyelerine kadar düşmüştür. Eğitimin doğruluk değerleri ise sırasıyla 10 epokta 0,4154 ve 0,2441; 25 epokta 0,4676 ve 0,2033; 50 epokta 0,7420 ve 0,4151; 100 epokta ise 0,9578 ve 0,7479 olarak elde edilmiştir.

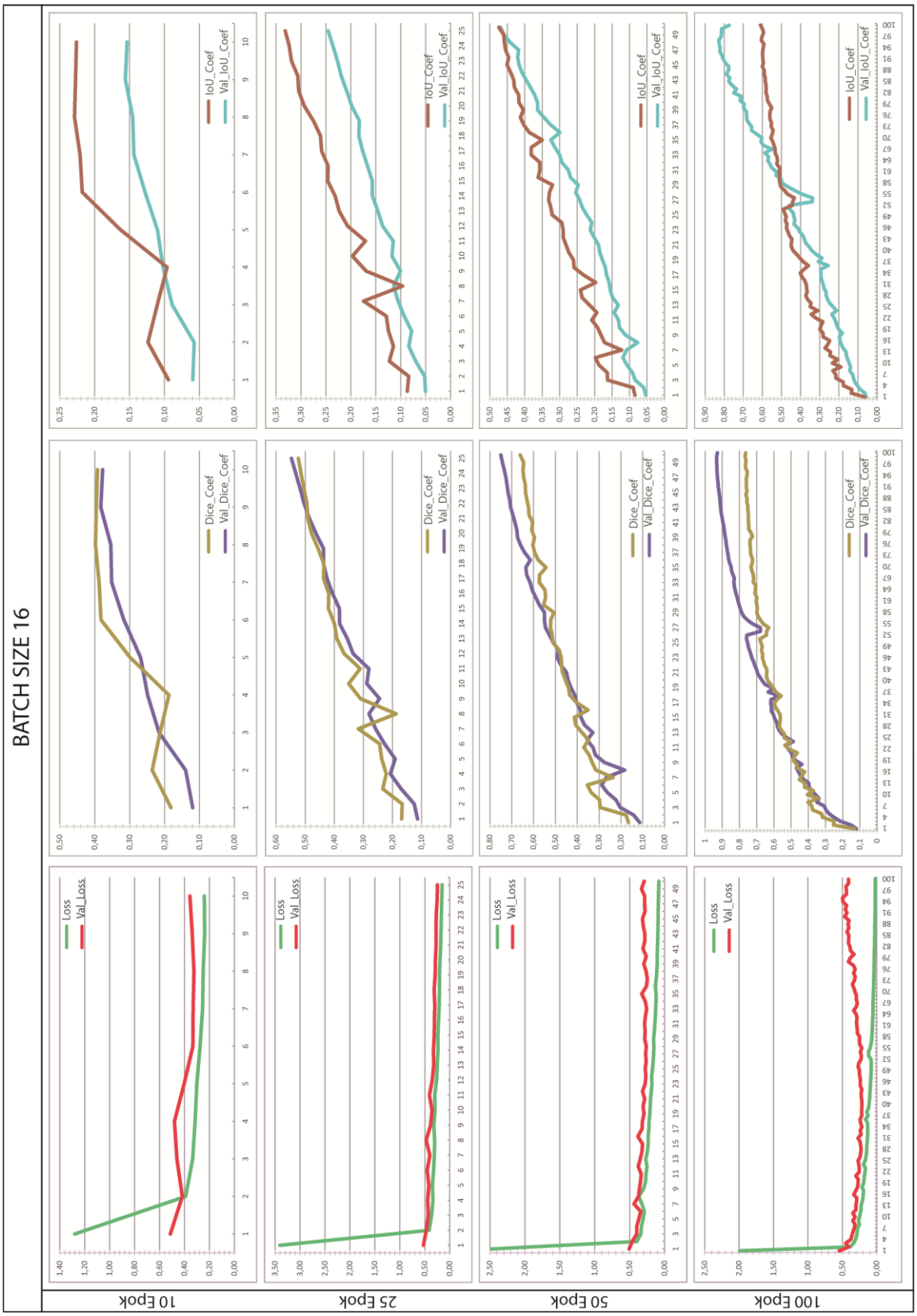
Epokların sayısı arttıkça, eğitimin ve doğrulamanın doğruluğu artarken, kayıp azalmıştır. Doğruluk artışı ve kayıp azalması başlangıçta hızla değişmiş, daha sonra yavaş yavaş ve kademeli olarak kararlı hale gelmiştir. Doğrulama kaybı değeri 0,32'den sonra stabil hale gelmeye başladı. Eğitim alanlarından elde edilen ağ modelinin, eğitim alanlarındaki binaları tanımlamak için %25 civarında bir doğruluğa ulaşabildiği görülmektedir.

Tüm eğitim süreleri için loss eğrisi zamanla azalmaktadır. Bu, modelin eğitim verileri üzerinde daha iyi performans gösterdiğini ifade etmektedir.
100 Epok ve 50 Epok eğitim süreleri için loss eğrisi daha hızlı azalmaktadır. Aynı epoklardaki eğitim süreleri için Dice Coef ve IoU Coef eğrisi daha hızlı artmaktadır. Bu, eğitim sürelerinin bina segmentasyonu için daha uygun olduğunu gösterir.

Batch boyutu 16 optimizasyonu incelendiğinde, en düşük doğruluk 10 epokta iken, en yüksek doğruluk 100 epokta gözükmektedir. Kayıp fonksiyonu 10 epok sonunda 0,2413; 25 epokta 0,1547; 50epokta 0,0809 ve 100 epokta 0,0245 seviyelerine kadar düşmüştür. Eğitimin doğruluk değerleri ise sırasıyla 10 epokta 0,3773 ve 0,1536; 25 epokta 0,5471 ve 0,2453; 50 epokta 0,7501 ve 0,4698; 100 epokta ise 0,9286 ve 0,7742 olarak elde edilmiştir. Batch boyutu 16'nın kullanıldığı optimizasyon sürecinin incelenmesi, modelin performansındaki belirgin gelişmeleri gözler önüne sermektedir. Bu süreçte, doğruluk değerleri ve kayıp fonksiyonu üzerinde dikkate değer değişimler kaydedilmiştir. İlk 10 epokta en düşük doğruluk değerleri gözlemlenirken, eğitim ilerledikçe bu değerlerde artan bir trend izlenmiştir. Özellikle, 100 epok sonunda elde edilen en yüksek doğruluk değeri, optimizasyon sürecinin başarısını vurgulamaktadır. Aynı şekilde, kayıp fonksiyonu da eğitim sürecinin ilerlemesiyle giderek düşmüş, 100 epokta minimum seviyeye ulaşmıştır. Bu durum, modelin veriye uyumu ve genel performansının sürekli olarak arttığını işaret etmektedir.



Şekil 4.1. Küme boyutu 8 için 10, 25, 50 ve 100 epoktaki Kayıp, Dice ve IoU grafikleri



Şekil 4.2. Küme boyutu 16 için 10,25,50 ve 100 epoktaki Kayıp, Dice ve IoU grafikleri

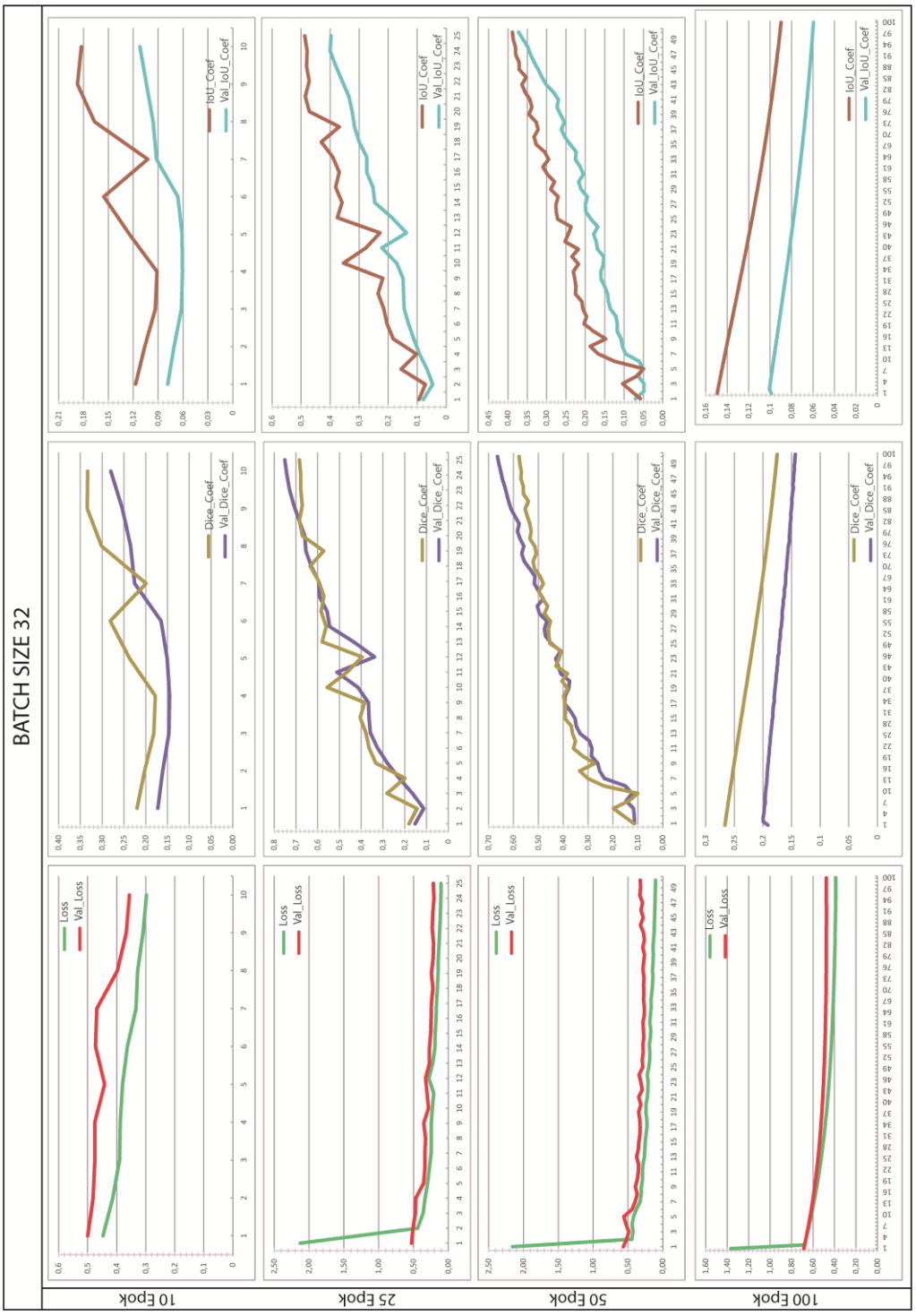
Batch boyutu 32 optimizasyonu için en düşük doğruluk 100 epokta iken, en yüksek doğruluk 25 epokta gözükmektedir. Kayıp fonksiyonu 10 epok sonunda 0,2976; 25 epokta 0,1044; 50epokta 0,1036 seviyelerine kadar düşmüştür. Kayıp fonksiyonu 10 epoktan başlayarak 50 epokta kadar düşmüş, ancak 50 epokun sonunda bir artış yaşanmıştır. Bu durum, istenmeyen bir durumu işaret etmektedir çünkü 100 epokta kaydedilen kayıp fonksiyonunun 50 epoktaki değere kıyasla %28 oranında arttığı görülmekte ve 0,3862 değerine ulaşmaktadır. Bu, modelin bazı veri noktalarında kötüleştiği veya aşırı uyum sağladığı anlamına gelebilir. Eğitimin doğruluk değerleri ise sırasıyla 10 epokta 0,2800 ve 0,1120; 25 epokta 0,7510 ve 0,3968; 50 epokta 0,6644 ve 0,3724; 100 epokta ise 0,1435 ve 0,0596 olarak elde edilmiştir. 25 epok sonunda hem Dice hem de IoU doğruluk metriklerinde bir azalma söz konusu olmuştur.

Her üç batch boyutu birbirlerine göre sonuçları değerlendirildiğinde, Batch boyutu 8 için, kayıp fonksiyonu eğitim sürecinin ilerlemesiyle istikrarlı bir şekilde azalmış ve 100 epokta minimum bir değere ulaşmıştır. Aynı şekilde, doğruluk değerleri de zamanla artmış ve 100 epokta oldukça yüksek bir seviyeye ulaşmıştır. Bu durum, küçük batch boyutlarının eğitim sürecini iyi bir şekilde optimize ettiğini ve modelin verilere daha iyi uyum sağladığını göstermektedir. Batch boyutu 16 için, kayıp fonksiyonu ve doğruluk değerleri batch boyutu 8'e benzer bir eğilim göstermektedir. Ancak, 16'nın biraz daha büyük bir batch boyutu olması nedeniyle, kayıp fonksiyonunun ve doğruluk değerlerinin 8'e kıyasla biraz daha hızlı bir şekilde iyileştiği görülmektedir. Bununla birlikte, genel olarak, batch boyutu 16'nın da eğitim sürecini başarılı bir şekilde optimize ettiği ve modelin iyi bir performans sergilediği söylenebilir. Batch boyutu 32 için ise, eğitim sürecinin bir noktasında beklenmedik bir değişim gözlenmiştir. 100 epokta kaydedilen kayıp fonksiyonunun, 50 epoğa kıyasla artış göstermesi, bu batch boyutunun modelin performansını olumsuz yönde etkilediğini göstermektedir. Doğruluk değerlerinde de benzer bir eğilim gözlenmiştir. Bu durum, batch boyutu 32'nin eğitim sürecini istenmeyen bir şekilde etkilediğini ve modelin verilere daha az uyum sağladığını işaret etmektedir. Bu nedenle, batch boyutu 32'nin performansı diğer iki batch boyutuna kıyasla daha düşüktür. En iyi sonuçların batch boyutu 16'da olduğu görülmektedir.

Model eğitiminin tamamlanması her bir batch boyutu ve epoktaki yaklaşık süre durumları Çizelge 4.1'de sunulmuştur. 10 epokta yaklaşık 2 saat süren eğitim, 100 epok olduğunda yaklaşık 1 gün kadar sürmektedir. Epok sayısı arttıkça modelin eğitimi için geçen süre de artmaktadır.

Çizelge 4.1. 8,16, 32 batch boyutlarının epoklara ait model eğitimi için gereken süreler

Süreler	*10 epok*	*25 epok*	*50 epok*	*100 epok*
BS_8	2 sa 01 dk 30 sn	5 sa 39 dk 54 sn	11 sa 12 dk 15 sn	22 sa 2 dk 4 sn
BS_16	2 sa 13 dk 19 sn	5 sa 28 dk 37 sn	11 sa12 dk 4 sn	22 sa 39 dk 51 sn
BS_32	2 sa 27 dk 50 sn	7 sa 0 dk 52 sn	11 sa 42 dk 3 sn	22 sa 47 dk 41 sn



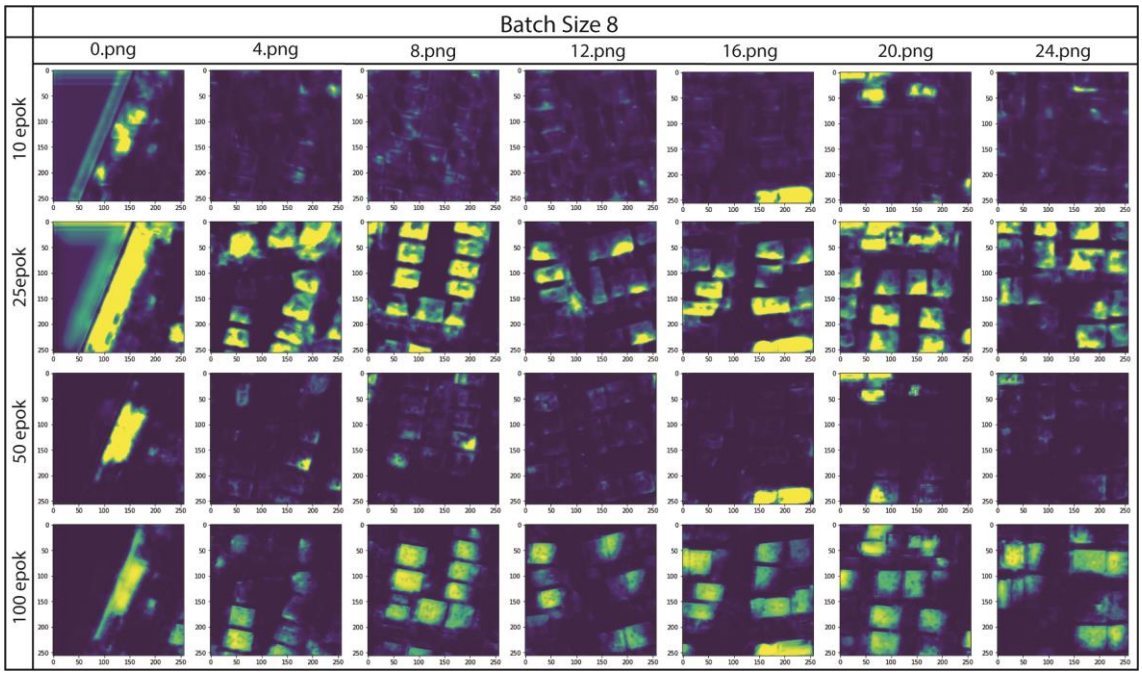
Şekil 4.3. Batch boyutu 32 için 10,25,50 ve 100 epoktaki Kayıp, Dice ve IoU grafikleri

Çizelge 4.2'de doğruluk, hassasiyet, duyarlılık, F1 skor, jaccard ve dice metriklerine ait değerler verilmiştir. İlk olarak, doğruluk (accuracy) değerlerine bakıldığında, batch boyutu 16 için 100 epokta en yüksek doğruluk elde edilmiştir. Benzer şekilde batch boyutu 32 için 50 epokta en yüksek doğruluk elde edilmiştir. Batch boyutu 8 içinse diğerlerine kıyasla daha düşük doğruluk değerleri kaydedilmiştir. Bu, batch boyutlarının farklı eğitim süreçleri için farklı etkileri olduğunu göstermektedir. Hassasiyet, duyarlılık ve F1 skoru gibi performans metrikleri incelendiğinde, batch boyutu 16 için genellikle diğerlerine kıyasla daha yüksek değerler elde edilmiştir. Özellikle, batch boyutu 16 için 100 epokta yüksek hassasiyet, duyarlılık ve F1 skoru gözlemlenmektedir. Ancak, batch boyutu 8 ve 32 için performans metrikleri daha değişkenlik göstermektedir. Jaccard ve Dice skorları yönünden bakıldığında, batch boyutu 16 için genellikle diğerlerine kıyasla daha yüksek değerler görülmektedir. Ancak, tüm batch boyutları için bu skorlar oldukça yüksek ve benzerdir. Genel olarak, bu sonuçlar batch boyutlarının eğitim süreci üzerinde önemli bir etkiye sahip olduğunu ve farklı batch boyutlarının farklı performans sonuçlarına yol açabileceğini göstermektedir. En uygun batch boyutunu seçmek, belirli bir problem ve veri seti için dikkatlice düşünülmesi gereken bir karardır.

Çizelge 4.2 Doğruluk metrikleri

Batch size		**Genel Doğruluk**	**Kesinlik**	**Duyarlılık**	**F1 Skoru**	**Jaccard**	**Dice**
BS_8	10	0,502631	0,601732	0,337787	0,432373	0,158434	0,305552
	25	0,593404	0,662392	0,488631	0,561674	0,198814	0,442320
	50	0,602325	0,668999	0,501317	0,571836	0,168809	0,693452
	100	0,553836	0,635477	0,424723	0,508284	0,587050	0,863891
BS_16	10	0,486314	0,591393	0,308304	0,404640	0,118396	0,28093
	25	0,564452	0,642774	0,441602	0,522246	0,208807	0,48359
	50	0,645208	0,626379	0,499962	0,586386	0,429015	0,72215
	100	0,822340	0,754763	0,571154	0,647338	0,818911	0,91568
BS_32	10	0,582340	0,654763	0,471154	0,547338	0,088902	0,197786
	25	0,527099	0,618177	0,378216	0,467142	0,348566	0,670640
	50	0,632379	0,689801	0,548515	0,610568	0,328738	0,639558
	100	0,563444	0,642249	0,4393	0,519962	0,068862	0,156150

Şekil 4.4'te batch boyutu 8 için uygulanan ve test verisinden seçilen 7 adet 256*256 boyutunda tahmin edilen görüntülerin, 10 epok, 25epok, 50epok ve 100 epoktaki görselleri verilmiştir. Çizelge4.3'te ise bu görüntülerin 100 epoktaki metrik değerleri ifade edilmiştir. 100 epokta elde edilen tahminler de oldukça net ve keskindir. 25epok için elde edilen tahminlere göre çok az bir fark vardır. Epok sayısı 10 ve 50 olduğunda tahminlerde bulanıklık ve gürültü gözlemlenmektedir, ancak 100 epok için elde edilen tahmin edilen görüntüler de bu durum daha azdır. Epok sayısının arttıkça, tahmin edilen görüntülerin daha iyi olması beklenir. 10 epoktan 25 epoğa geçişte iyileşme varken, 50 epokta görüntüleri tahmin etmek neredeyse zor olmuştur. Fakat 100 epokta binaların konumları belirgin olarak görülmektedir. Sadece, keskin bir sınırları bulunmamaktadır. 0.png görüntüsü tüm epoklar da istenilen sonucu vermemiştir. 4.png isimli görüntüde 10 epokta tahmin etme oldukça yetersiz iken, 25 epokta binaları tahmin etmiş fakat anlamlı bir sonuç üretmemiştir. 50 epokta model eğitiminde hiçbir görüntü anlamlı değildir. 4.png isimli görüntünün 100 epoktaki durumu, sol alt kısımdaki binalar belirlenmiş, fakat üst tarafındaki binalar belirlenememiştir. Bunun temel sebebi de bu görüntünün üst taraflarında gölge bulunmasından dolayı RGB görüntüde binalar net belirgin değildir. Benzer durum 24.png isimli görüntüde geçerlidir. 8,16 ve 20 numaralı görüntüler diğerlerine kıyasla daha anlamlı sonuçlar üretmiştir.



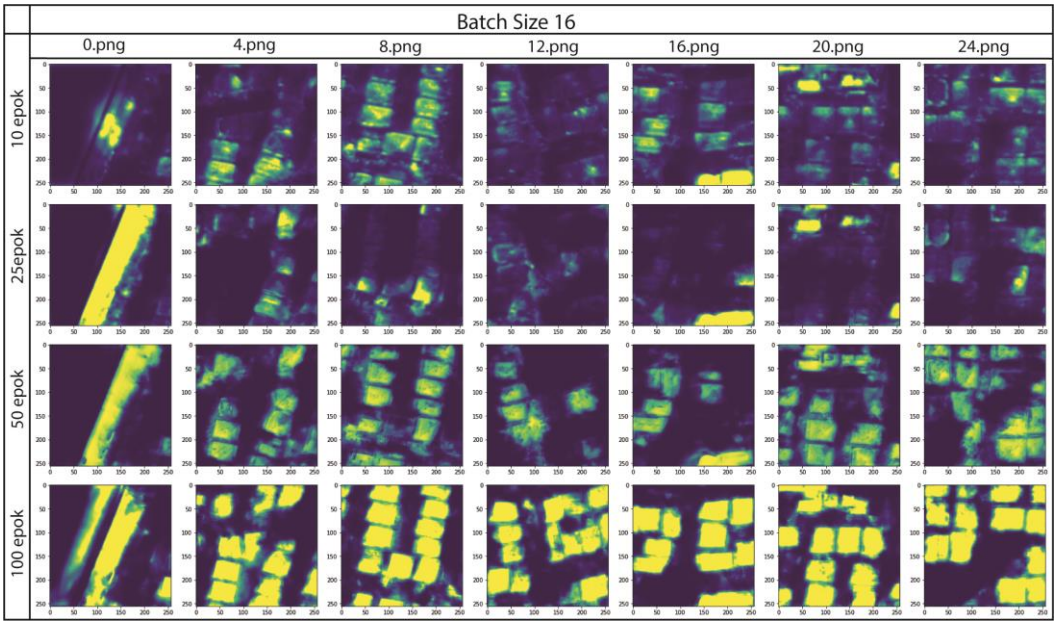
Şekil 4.4. Batch boyutu 8 için üretilen tahmin edilen görüntüleri

Batch boyutu 8 ve epok sayısı 100 olduğunda genel doğruluk değeri %63,54 iken, dice benzerlik oranı %77,39 olarak elde edilmiştir. Çizelgedeki her bir görüntüye ait metrik incelendiğinde, Dice katsayısının ve doğruluk değerinin en yüksek 8 numaralı görüntüde, ardından 12 ve 16 numaralı görüntülerde olduğu görülmektedir.

Çizelge 4.3. Batch boyutu 8'nun 100 için üretilen tahmin edilen görüntülerin metrik değerleri

	Görüntü Numarası	Genel Doğruluk	Kesinlik	Duyarlılık	F1 Skoru	Jaccard Benzerliği	Dice Katsayısı
Batch Size 8_100epok	**0.png**	0,545506	0,698879	0,569136	0,62737	0,658823	0,715709
	4.png	0,532569	0,654649	0,472465	0,548833	0,718483	0,776059
	8.png	0,589194	0,625484	0,401238	0,488872	0,748669	0,795704
	12.png	0,554963	0,635967	0,427592	0,511367	0,729009	0,789469
	16.png	0,564139	0,64213	0,442682	0,524072	0,728529	0,781928
	20.png	0,516274	0,610484	0,361954	0,45446	0,748385	0,773878
	24.png	0,55834	0,63823	0,433166	0,516073	0,689066	0,746547

Görüntüler, batch boyutu 16 olarak ayarlanmış bir CNN modelinin eğitim sürecinde elde edilmiştir. Batch boyutu 16'da tahmin edilen görüntüler, batch boyutu 8'e kıyasla daha güzel sonuçlar vermiştir. 10, 25 ve 50 epokta bina tahmini gerçekleşse dahi eksiklikler bulunmaktadır. Fakat, 100 epoktaki görsellerde 0 ve 4 numaralı görüntüler haricinde tüm binaların bulunması umut vericidir. 100 epoktaki eğitimden sonra düşük bir hata fonksiyonu elde etmiştir. Bu, modelin eğitim verilerindeki görüntüleri iyi öğrendiğini göstermektedir. Çizelge 4.4'te 100 epoktaki metrik değerler ifade edilmiştir.



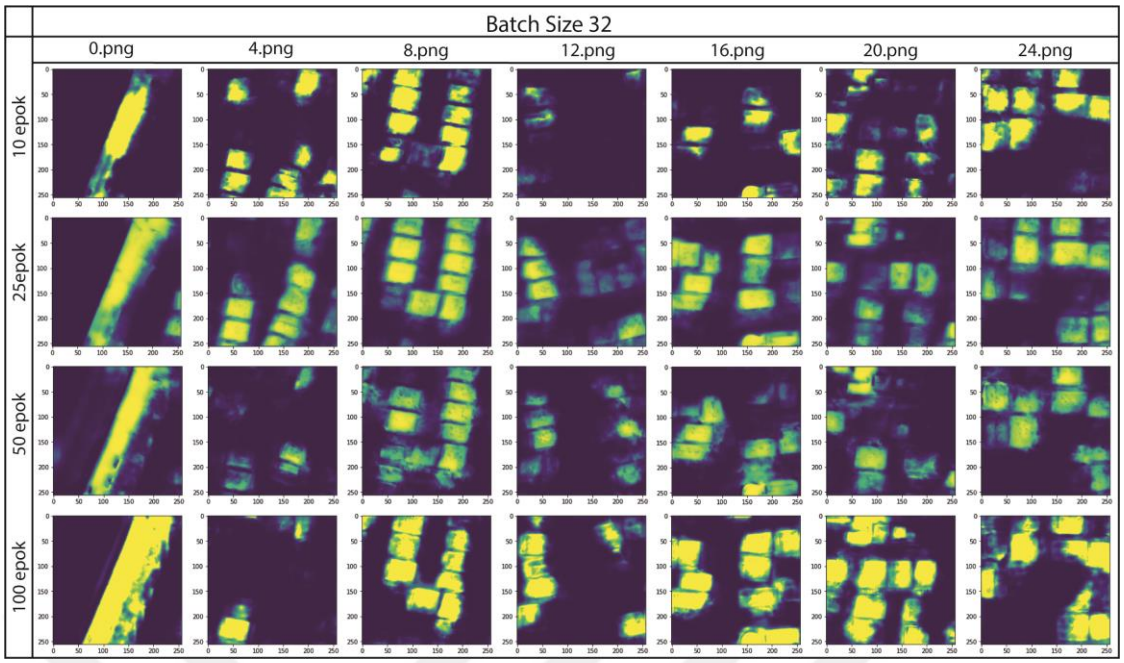
Şekil 4.5. Batch boyutu 16 için üretilen tahmin edilen görüntüleri

100 epoktaki dice katsayısının ortalaması, %76,22 ve doğruluğu ise, %65,47 olarak elde edilmiştir. Çizelgedeki, her bir görüntüye ait metrik değerler incelendiğinde 16 numaralı görüntüdeki doğruluk değeri en yüksek iken, dice katsayısının en iyi olduğu görüntü 8 numaradır. Tüm görüntüler için 0.56 ile 0.66 arasında değişen bir doğruluk görülmektedir. Bu, modelin genel olarak doğru tahminler yaptığını ancak bazı iyileştirmelerin yapılması gerektiğini ifade etmektedir. Dice katsayısında ise genellikle 0.72 ile 0.81 arasında değiştiği görülmektedir. Bu durum da modelin ikili sınıflandırma problemleri için iyi bir performans sergilediği anlamına gelmektedir.

Çizelge 4.4. Batch boyutu 16'nun 100 için üretilen tahmin edilen görüntülerin metrik değerleri

	Görüntü Numarası	Genel Doğruluk	Kesinlik	Duyarlılık	F1 Skoru	Jaccard Benzerliği	Dice Katsayısı
Batch Size 16_100epok	**0.png**	0,579814	0,652766	0,468057	0,545192	0,629153	0,728518
	4.png	0,589277	0,659253	0,483132	0,557617	0,688581	0,780783
	8.png	0,560233	0,639501	0,436281	0,518696	0,758828	0,814826
	12.png	0,557363	0,637574	0,431555	0,514715	0,798813	0,801411
	16.png	0,612203	0,675184	0,518922	0,586829	0,758642	0,724923
	20.png	0,564182	0,642159	0,442753	0,524131	0,798669	0,811308
	24.png	0,65879	0,708518	0,588604	0,643018	0,669076	0,773982

Aynı eğitim görüntüleri, batch boyutu 32 olarak ayarlanmış şekilde tekrar eğitilmiştir. Batch boyutu 32'de tahmin edilen görüntüler, batch boyutu 16'dan daha iyi sonuçlar üretmesi bekleniyordu. Fakat, istenilen durum olmadı. Batch boyutu 32 ve epok sayısı 25 durumunun batch boyutu 16 ile karşılaştırıldığında daha anlamlı sonuç ürettiği görülmektedir. Fakat 100 epoktaki sonuçların batch boyutu 16'da daha iyi olduğu tespit edilmiştir. Batch boyutu 32'de 25 epok ve 50 epok sonuçları birbirine yakındır. 0 numaralı görüntüden kaynaklı anlamlı sonuç üretememiştir. Fakat 25 epokta bulması gereken üç adet binadan iki tanesinin doğru şekilde tespit edildiği görülmüştür. Diğer epoklar da bu görüntü için bina tespit edilememiştir. 4 numaralı görüntü için en iyi sonuç batch boyutu 32'de elde edilmiştir. Çizelge 4.5'te 100 epok için metrik değerleri bulunmaktadır.



Şekil 4.6. Batch boyutu 32 için üretilen tahmin edilen görüntüleri

100 epoktaki tüm görüntülerin dice katsayısının ortalaması, %11,67 ve doğruluğu ise, %47,77 olarak elde edilmiştir. Ortalama doğruluk değeri yaklaşık olarak 0.47 bu da modelin tahminlerinin yaklaşık olarak yarısının doğru olduğunu gösterir. Ancak, %50'nin altında bir doğruluk genellikle düşük performans olarak kabul edilir. Çizelgedeki görüntüler için metrikler yorumlandığında, ortalama hassasiyet değeri yaklaşık olarak %55'tir. Bu, modelin pozitif olarak tahmin ettiği örneklerin yarısının gerçekten pozitif olduğunu gösterir. Ortalama duyarlılık değeri yaklaşık olarak 0.45 ve Ortalama F1 skoru yaklaşık olarak 0.51'dir. Ortalama Dice katsayısı değeri yaklaşık olarak 0.21'dir. Dice katsayısı, tahmin edilen ve gerçek etiketler arasındaki benzerliği ölçer. Yüksek bir Dice katsayısı, tahminlerin gerçek etiketlere yakın olduğunu gösterir. Fakat burada oldukça küçük olduğu ve tahmin edilen görüntünün gerçek değer benzemediği açıkça görülmektedir.

Çizelge 4.5. Batch boyutu 32'nin 100 için üretilen tahmin edilen görüntülerin metrik değerleri

	Görüntü Numarası	Genel Doğruluk	Kesinlik	Duyarlılık	F1 Skoru	Jaccard Benzerliği	Dice Katsayısı
Batch Size 32_100epok	**0.png**	0,47186	0,547351	0,455244	0,534562	0,068658	0,155677
	4.png	0,53095	0,588441	0,547442	0,609899	0,098709	0,198627
	8.png	0,430302	0,519631	0,386137	0,475781	0,098663	0,259329
	12.png	0,487354	0,557931	0,480083	0,55511	0,118519	0,280443
	16.png	0,473142	0,548222	0,457319	0,536288	0,158503	0,279972
	20.png	0,456824	0,537213	0,430667	0,513965	0,088746	0,186009
	24.png	0,473617	0,548545	0,458086	0,536926	0,069154	0,13771

Batch boyutu, modelin her güncellemede kaç görüntü işleyeceğini belirler. Batch boyutu ne kadar yüksek olursa, model o kadar hızlı öğrenebilir. Ancak, batch boyutu çok yüksekse, model eğitim verilerindeki gürültüye karşı daha hassas hale gelebilir. Batch boyutlarındaki sonuçlar incelendiğinde en iyi değerlerin Batch boyutu 16'da olduğu görülmektedir.

F1 skoru, hassasiyet ve duyarlılık arasındaki dengeyi sağlar, her üç batch boyutu için değerler modelin dengeleme performansının orta düzeyde olduğunu göstermektedir.

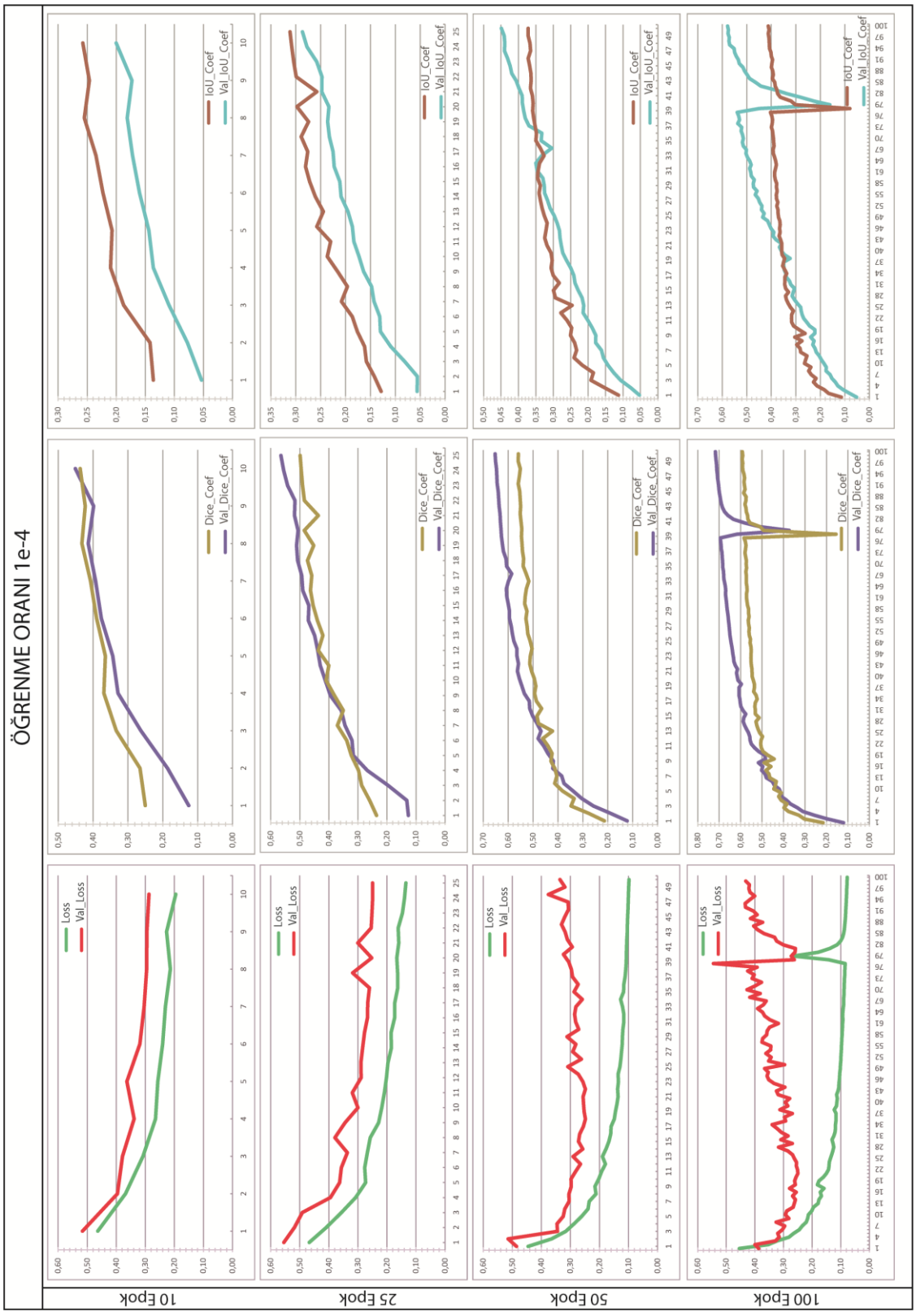
Epok sayısı artıkça, doğruluk, jaccard ve dice katsayılarında da artış olduğu görülmektedir. Batch boyutu 32 haricinde geneli bu durumu sağlamaktadır.

4.2. Bina için Farklı Öğrenme Oranlarının Sonuç Değerlendirmesi

Bu çalışmanın devamında, bina yapıları için farklı öğrenme oranlarının sonuçlarını değerlendirmek amacıyla batch boyutunda olduğu gibi U-Net mimarisi benimsenmiştir. Optimal model parametrelerini belirlemek için adam algoritması tercih edilmiş ve batch boyutu sabit olarak 16 olarak belirlenmiştir. Bu mimarinin kullanılması, bina yapıları için hassas sonuçlar elde etme potansiyelini arttırmaktadır. Ayrıca, adam algoritmasının optimizasyon yöntemi olarak seçilmesi, modelin hızlı bir şekilde eğitilmesine ve daha iyi sonuçlar elde edilmesine olanak tanımaktadır. Batch boyutunun sabit olarak 16 seçilmesi ise, eğitim sürecinin stabilitesini sağlamak ve bellek kullanımını optimize etmek için yapılmış bir tercihtir. Bu parametreler altında yapılan çalışmada, bina yapıları için farklı öğrenme oranlarının etkisini kapsamlı bir şekilde değerlendireceği öngörülmektedir. Özellikle, 1e-4 ve 1e-3 öğrenme oranları üzerinde durularak elde edilen sonuçlar detaylı bir şekilde değerlendirilmiştir. Şekil4.7'de öğrenme oranı 1e-4 için kayıp, dice ve IoU grafiğine yer verilmiştir. Batch size 16 da öğrenme oranı 1e-3 olarak kullanıldığından tekrar olmaması açısından şekil ve çizelgeleri eklenmemiştir. İhtiyaç olması durumunda Şekil4.5 ve Çizelge 4.4'te veriler incelenebilir.

Öğrenme oranı 1e-4 grafikleri incelendiğinde, en düşük doğruluk 10 epokta iken, epok sayısındaki artmalar, doğruluk oranını da arttırmıştır. 100 epokta ise 10 epoktan daha yüksek doğruluğa erişmiştir. Kayıp fonksiyonu 10 epok sonunda 0,1952; 25 epokta 0,1346; 50epokta 0,0983 ve 100 epokta 0,0775 seviyelerine kadar düşmüştür. Eğitimin dice ve IoU metrik değerleri ise sırasıyla 10 epokta 0,4509 ve 0,2002; 25 epokta 0,5650 ve 0,2865; 50 epokta 0,6527 ve 0,4489; 100 epokta ise 0,7177 ve 0,5780 olarak elde edilmiştir. Doğruluk artışı ve kayıp azalması başlangıçta hızla değişmiş, daha sonra yavaş

yavaş ve kademeli olarak kararlı hale gelmiştir. Fakat 79. Epokta kayıp değeri düşmesi gerekirken artmıştır. Dice ve IoU da ise tam tersi gözlemlenmiştir. Doğruluk değerleri artması gerekirken, 79. Epokta ani bir düşüş yaşanmıştır. Bu durumun sebepleri arasında; kullanılan donanım yetersiz olması ya da aşırı öğrenme durumu söz konusu olabilir. Donanımın yetersiz olması durumda, sinir ağı zamanında ve doğru şekilde güncellenemeyebilir ve bu da doğruluk değerlerinde bir düşüşe neden olabilir. Sinir ağı, eğitim verisini aşırı öğrenmesi durumunda ise, yeni veriler için genelleme yeteneğini kaybetmiş olabilir. Bu durumda, eğitim verilerinde yer alan kalıpları ezberlemiş ve yeni verilere uygulanamamaktadır. Çizelge 4.6'da öğrenme oranı 1e-4 ve 1e-3 için eğitimdeki geçen süreler ifade edilmiştir. En kısa eğitim süresi, 10 epok ve "1e-3" öğrenme oranı kombinasyonunda gözlemlenmiştir (2 saat 13 dakika 19 saniye). 100 epoktaki sürelere bakıldığında ise 1e-4 için yaklaşık 30 saat sürerken, 1e-3 için yaklaşık 1 gün kadar sürmektedir. Epok sayısı arttıkça modelin eğitimi için geçen süre de artmaktadır. Bu durum da epok sayısı ve öğrenme oranının eğitim süresi üzerinde önemli bir etkiye sahip olduğunu göstermektedir. 1e-4" öğrenme oranı, "1e-3" öğrenme oranına göre daha uzun eğitim sürelerine yol açmaktadır. Bunun sebebi, "1e-4" öğrenme oranının modelin parametrelerini daha küçük adımlarla güncellemesi ve bu nedenle her epokta daha az ilerleme kaydetmesidir.



Şekil 4.7. Öğrenme oranı 1e-4 için 10,25,50 ve 100 epoktaki Kayıp, Dice ve IoU grafikleri

Çizelge 4.6. Farklı öğrenme oranları için model eğitiminde geçen süre

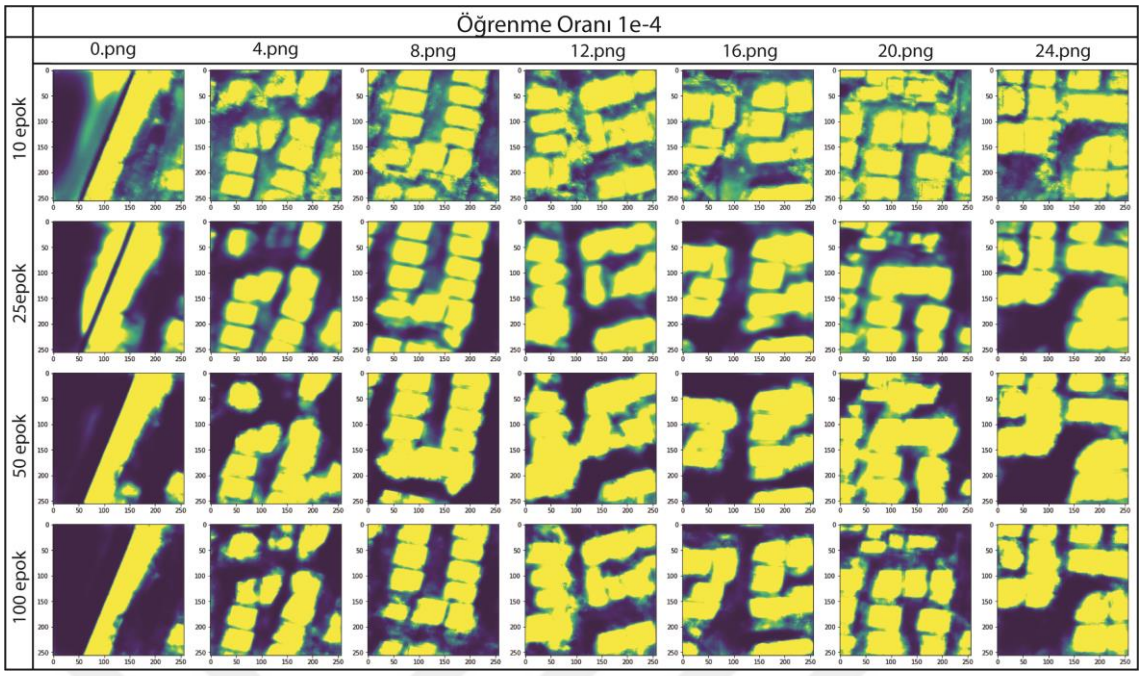
Süreler	***10 epok***	***25 epok***	***50 epok***	***100 epok***
1e-4	2 sa 14 dk 42 sn	5 sa 43 dk 29 sn	11 sa 36 dk 44 sn	30 sa 30 dk 32 sn
1e-3	2 sa13 dk 19 sn	5 sa 28 dk 37 sn	11 sa12 dk 4 sn	22 sa 39 dk 51 sn

Çizelge 4.7'de doğruluk, hassasiyet, duyarlılık, F1 skor, jaccard ve dice metriklerine ait değerler verilmiştir. %64,83 oranıyla en yüksek doğruluk, "1e-3" öğrenme oranında ve 100 epok kombinasyonunda gözlemlenmiştir. "1e-3" öğrenme oranı ve 100 epok kombinasyonunda hassasiyet, duyarlılık, F1 skoru, Jaccard ve Dice parametleri de en iyi değerlere ulaşmıştır. Genel olarak "1e-3" öğrenme oranının "1e-4" öğrenme oranına göre daha iyi performans gösterdiğini anlaşılmaktadır. Ayrıca, 100 epokluk eğitimin, daha az epoklu kombinasyonlara göre daha iyi sonuçlar verdiği tespit edilmiştir.

Çizelge 4.7. Öğrenme oranlarına ait metrikler

Batch Size		**Genel Doğruluk**	**Kesinlik**	**Duyarlılık**	**F1 Skoru**	**Jaccard**	**Dice**
LR_1e-4	10	0,556014	0,636841	0,428657	0,511748	0,148402	0,339343
	25	0,566365	0,644026	0,444749	0,524487	0,258440	0,528238
	50	0,605578	0,661968	0,553711	0,614859	0,428614	0,641082
	100	0,587138	0,657935	0,479213	0,554180	0,568457	0,712769
LR_1e-3	10	0,486314	0,591393	0,308304	0,404640	0,118396	0,28093
	25	0,564452	0,642774	0,441602	0,522246	0,208807	0,48359
	50	0,645208	0,626379	0,499962	0,586386	0,429015	0,72215
	100	0,822340	0,754763	0,571154	0,647338	0,818911	0,91568

10 epokluk eğitimde modelin performansı oldukça zayıftır. 25 epokluk eğitimde modelin performansı biraz daha artmıştır. 50 ve 100 epokluk eğitimlerde ise modelin performansının daha da arttığı ve daha az sayıda yanlış tahmin yaptığı gözlemlenmektedir. İstenilen duruma gelmese de olumlu sonuçlar üretmiştir. "1e-4" öğrenme oranında tahminler, görsel anlamda yorumlandığında 10 epok, 25 epok ve 50 epok sonuçları "1e-3" öğrenme oranına kıyasla daha iyidir. Çizelge 4.8'de "1e-4" öğrenme oranının 100 epok için metrik değerleri bulunmaktadır.



Şekil 4.8. 1e-4 öğrenme oranını sonucunda tahmin edilen görüntüler

100 epoktaki tüm görüntülerin dice katsayısının ortalaması, %64,37 ve doğruluğu ise, %59,58 olarak elde edilmiştir. Çizelgede doğruluk, hassasiyet, duyarlılık ve F1 skoru 0,55 ile 0,64 arasında değiştiği görülmektedir. Bu, modelin doğru tahminlerde bulunma şansının %54 ile %64 arasında olduğunu ifade etmektedir. Jacard ve Dice benzerliklerinin ortalaması ise sırasıyla %58 ve %67'dir. Tahmin edilen ve gerçek etiketler arasındaki benzerlik durumunun orta derecede örtüştüğünü göstermektedir. 0 numaralı görüntüde, model en düşük doğruluk, hassasiyet, duyarlılık ve F1 skoruna sahiptir. 20 numaralı görüntü de modelin performansı daha da iyi olduğu tespit edilmiştir. Dice katsayının en yüksek olduğu doğruluk bu görüntüdedir. Bazı görüntülerde diğerlerinden daha iyi performans gösterse de genel olarak orta derecede bir segmentasyon işlevi gerçekleştirilmiştir.

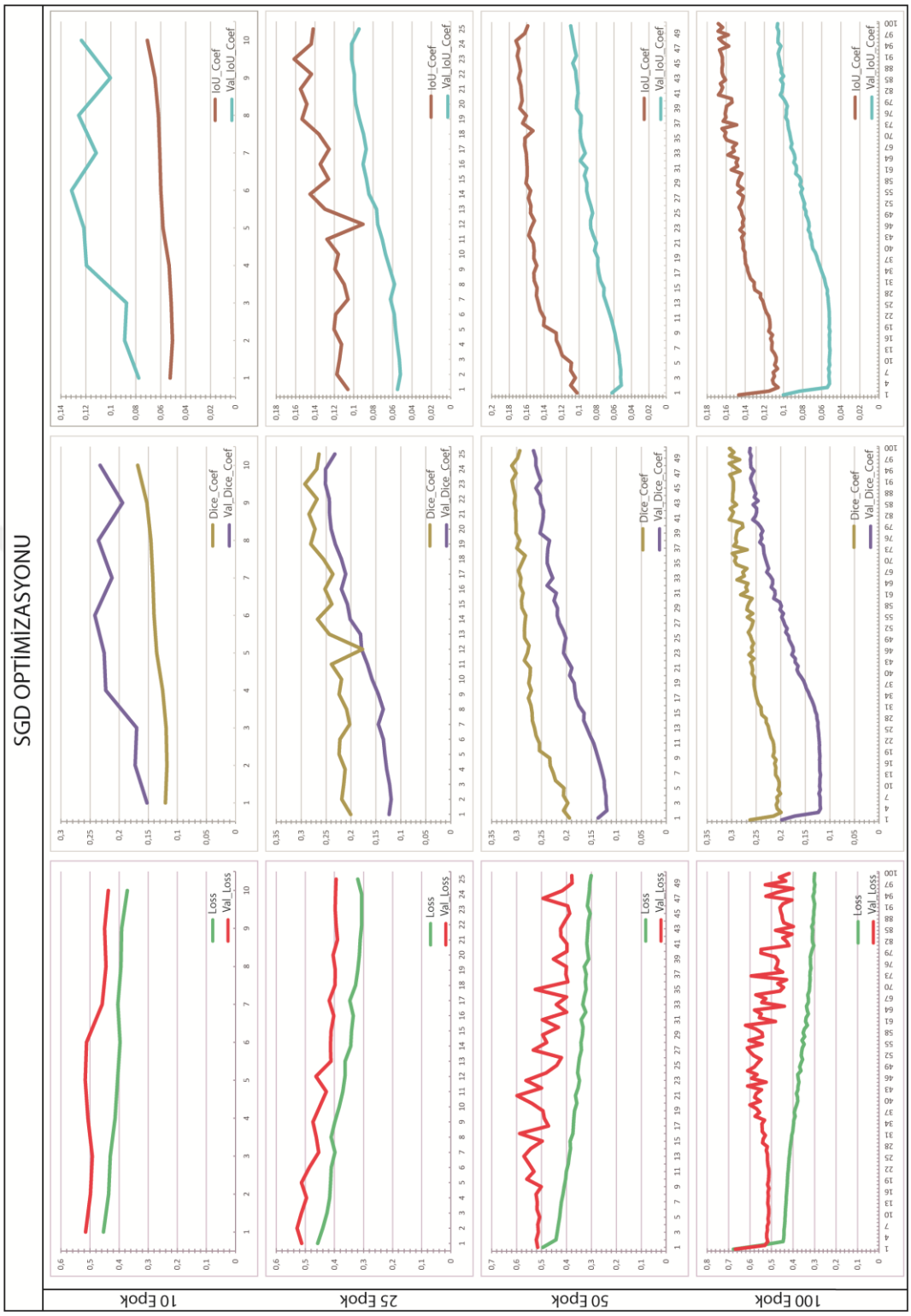
Çizelge 4.8. 1e-4 öğrenme oranının 100 epoktaki görüntüleri için metrik değerleri

	Görüntü Numarası	Genel Doğruluk	Kesinlik	Duyarlılık	F1 Skoru	Jaccard Benzerliği	Dice Katsayısı
LR_1e-4_100epok	**0.png**	0,541041	0,595663	0,462523	0,522049	0,568797	0,645518
	4.png	0,579235	0,65237	0,467128	0,544423	0,587963	0,700483
	8.png	0,559662	0,639117	0,435342	0,517906	0,588143	0,701911
	12.png	0,572668	0,6479	0,456551	0,53565	0,608453	0,657381
	16.png	0,574043	0,648834	0,458773	0,537496	0,618318	0,694403
	20.png	0,566833	0,643948	0,44708	0,527752	0,608324	0,704131
	24.png	0,641323	0,695866	0,562942	0,622386	0,588778	0,656426

Derin öğrenme modellerinde, öğrenme oranı, modelin ağırlıklarını ne kadar hızlı güncelleyeceğini belirleyen önemli bir hiperparametredir. Çünkü doğru öğrenme oranını belirlemek ve seçmek, modelin hem erken aşamalarında hem de son kısımlarında iyi performans göstermesi için kritik önem taşımaktadır. Tezin bu kısmında, "1e-3" ve "1e-4" öğrenme oranlarının, görüntü sınıflandırma problemindeki bir derin öğrenme modeli üzerindeki etkisi araştırıldı. Elde edilen bulgular doğrultusunda, "1e-3" öğrenme oranının daha iyi sonuç verdiği tespit edilmiştir.

4.3. Bina için Farklı Optimizasyon Yöntemlerinin Sonuç Değerlendirmesi

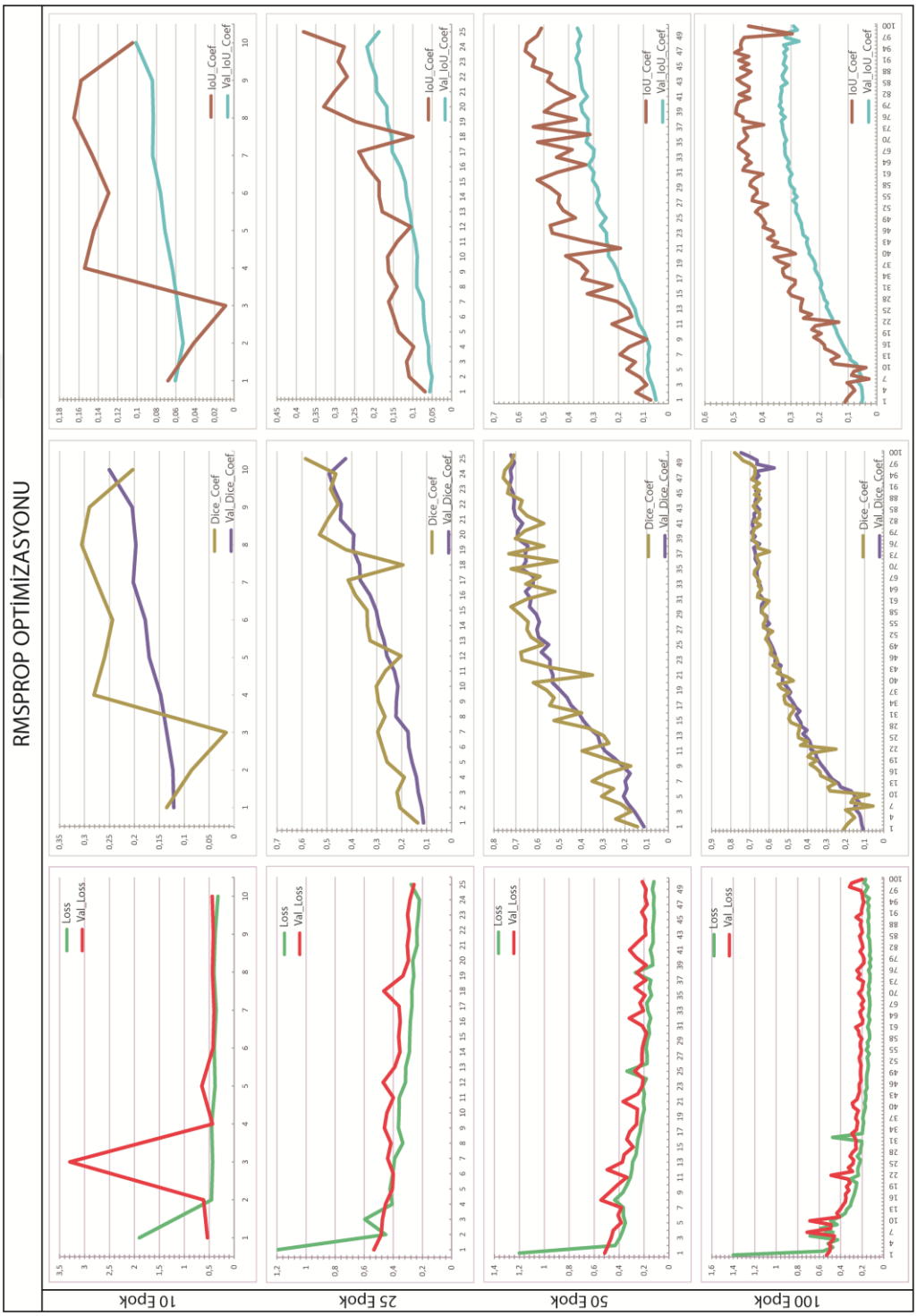
Bu tez çalışma kapsamında, binaların segmentasyonu için U-Net mimarisini ve çeşitli optimizasyon yöntemlerini kullanılmaktadır. Segmentasyon işleminde tercih edilen mimari, kentsel alanlardaki binaları hassasiyetle tespit etmek için uyarlanmıştır. Çalışmada, Adam, SGD, Adadelta, RMSprop, Nadam ve Adagrad olmak üzere altı farklı optimizasyon yöntemi karşılaştırılmıştır. Şekil4.9'da SGD, Şekil4.10'da RMSprop, Şekil4.11'de Adam, Şekil 4.12'de Nadam, Şekil4.13'te Adagrad, Şekil4.14'te Adadelta optimizasyonlarına ait kayıp ve doğruluk (Dice ve IoU) grafikleri sunulmuştur. Her bir parametrenin 10epok, 25epok, 50epok ve 100 epoktaki durumları incelenmiştir. Ayrıca Çizelge 4.9'da modelin eğitimi için harcanan süreleri ifade etmektedir.



Şekil 4.9. SGD için 10,25,50 ve 100 epoktaki Kayıp, Dice ve IoU grafikleri

Şekil 4.9'da SGD optimizasyonu için en düşük doğruluk 10 epokta iken, en yüksek doğruluk 100 epokta gözükmektedir. Kayıp fonksiyonu 10 epok sonunda 0,3720; 25 epokta 0,3199; 50epokta 0,3008 ve 100 epokta 0,2997 seviyelerine kadar düşmüştür. Eğitimin doğruluk değerleri ise sırasıyla 10 epokta 0,2927 ve 0,1236; 25 epokta 0,2643

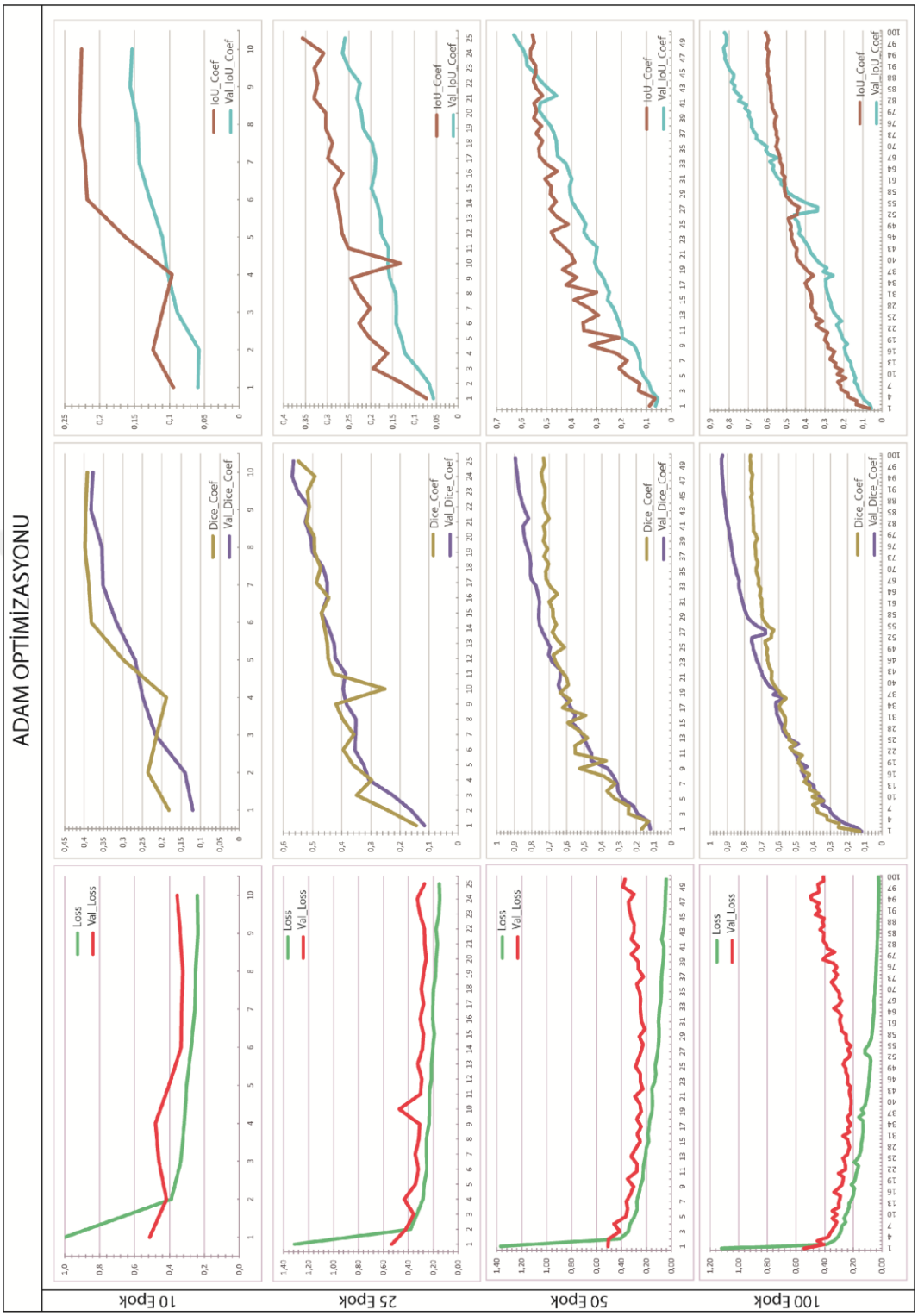
ve 0,1419; 50 epokta 0,2937 ve 0,1588; 100 epokta ise 0,3044 ve 0,1682 olarak elde edilmiştir. 25 epokta model performansının gözle görülür şekilde iyileştiği görülmektedir. 100 epokta model performansı düşük kaldığı görülmektedir.



Şekil 4.10. RMSprop için 10,25,50 ve 100 epoktaki Kayıp, Dice ve IoU grafikleri

Şekil 4.10’da RMSprop optimizasyonu sonucu elde edilen grafikler bulunmaktadır. Kayıp fonksiyonu 10 epok sonunda 0,3250; 25 epokta 0,2768; 50epokta 0,1240 ve 100 epokta 0,1687 seviyelerine kadar düşmüştür. Kayıp fonksiyonun değeri 0’a ne kadar yakın olursa o kadar iyi anlamındadır. Gözlemsel veriler, doğruluk ve kayıp fonksiyonlarının başlangıçta hızlı değişimler gösterirken ve belirli bir epoktan sonra sabit bir değere yaklaştığını göstermektedir. 100 epoktaki doğrulama kaybı değeri incelendiğinde 46. epoktan itibaren yani 0,15'ten sonra stabil hale gelmeye başlamıştır. 10 epokta 0,3250 civarında iken 100 epok sonunda 0,1687 olmuştur. Eğitimin doğruluk değerleri ise sırasıyla 10 epokta 0,2034 ve 0,1046; 25 epokta 0,5855 ve 0,3814; 50 epokta 0,7091 ve 0,5101; 100 epokta ise 0,7798 ve 0,5475 olarak elde edilmiştir.

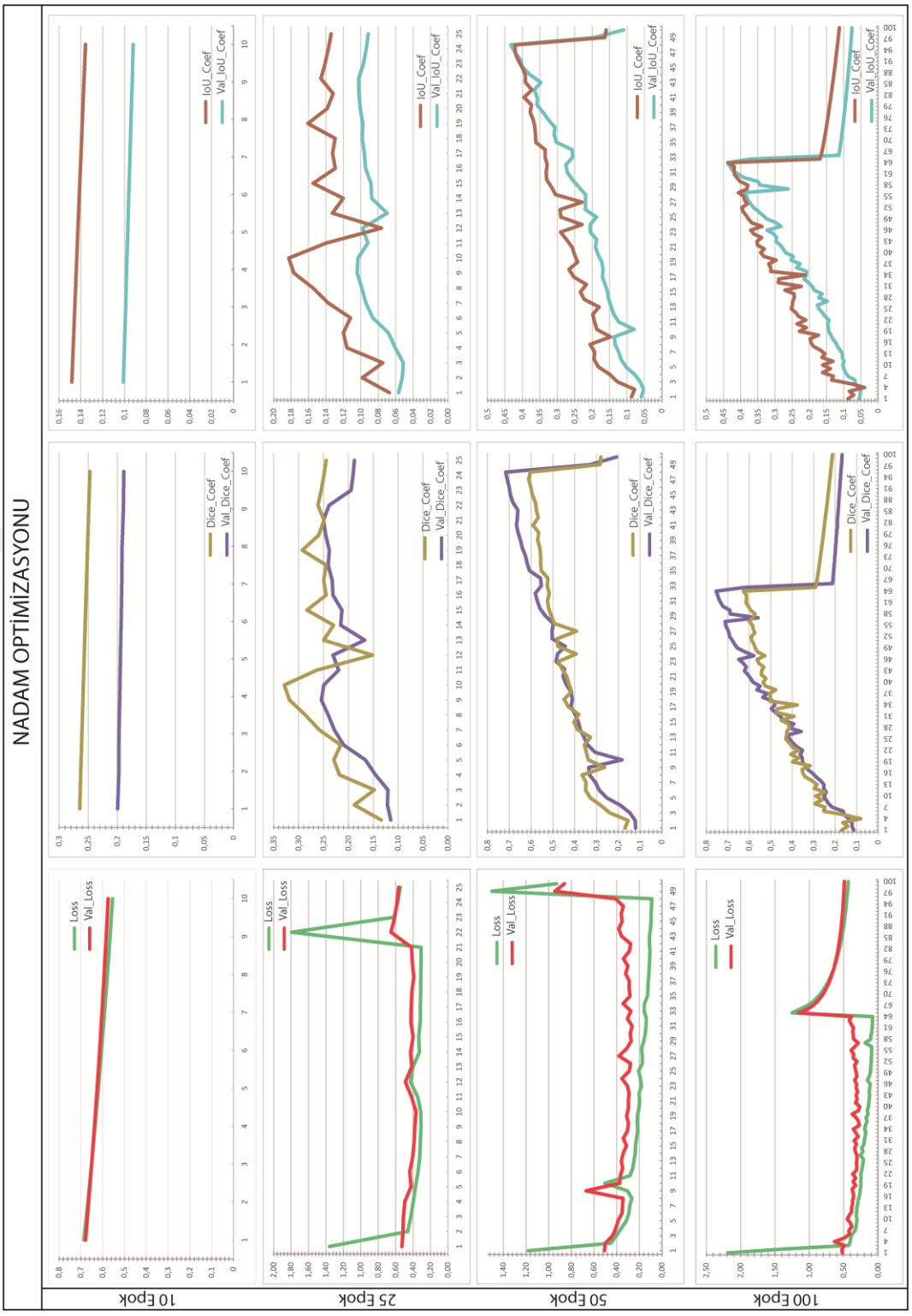
Şekil 4.11’de Adam optimizasyonu için en düşük doğruluk 10 epokta iken, en yüksek doğruluk 100 epokta gözükmektedir. Kayıp fonksiyonu 10 epok sonunda 0,241; 25 epokta 0,1508; 50epokta 0,0426 ve 100 epokta 0,0245 seviyelerine kadar düşmüştür. Modelin eğitim sürecinde kayıp fonksiyonunun azaldığını ve böylece modelin daha iyi performans göstermeye başladığı net bir şekilde görülmektedir. Daha düşük bir kayıp fonksiyonu, modelin daha doğru tahminler yapabildiği anlamına gelmektedir. Adam optimizasyonunda, 100 epokta kayıp fonksiyonunun 0,0245 seviyesine düşmesi, modelin oldukça iyi bir performans sergilediğini ifade ettiği dile getirilebilir. 100 epok grafiğinde doğruluk artışının ve kayıp azalmasının başlangıçta hızla değiştiği, daha sonrasında yavaş yavaş kademeli olarak 59 epoktan sonra kararlı hale geldiği görülmektedir. Eğitimin dice ve IoU doğruluk değerleri ise sırasıyla 10 epokta 0,3919 ve 0,2260; 25 epokta 0,5496 ve 0,3565; 50 epokta 0,7324 ve 0,5573; 100 epokta ise 0,8644 ve 0,6099 olarak elde edilmiştir. Modelin eğitimi ilerledikçe hem dice hem de IoU doğruluk değerlerinin arttığı anlaşılmıştır. Daha yüksek doğruluk değerleri, modelin daha iyi segmentasyon sonuçları ürettiğini ve tahminlerinin gerçek etiketlerle daha iyi örtüştüğünü işaret etmektedir. Kısacası bu değerler, modelin eğitiminin sonunda oldukça yüksek bir doğruluk seviyesine ulaştığını ve binaların segmentasyonunda başarılı olduğunu göstermektedir.



Şekil 4.11. Adam için 10,25,50 ve 100 epoktaki Kayıp, Dice ve IoU grafikleri

Şekil 4.12'de Nadam optimizasyonu kayıp ve doğruluk değerlerine ait grafikler sunulmuştur. Kayıp fonksiyonu 10 epok sonunda 0,5544; 25 epokta 0,5480; 50epokta 0,9307 ve 100 epokta 0,4338 seviyelerine kadar düşmüştür. Diğer optimizasyonlarda olduğu gibi kayıp fonksiyonu sürekli azalmamıştır. 50 epokta bir miktar artmıştır. 50

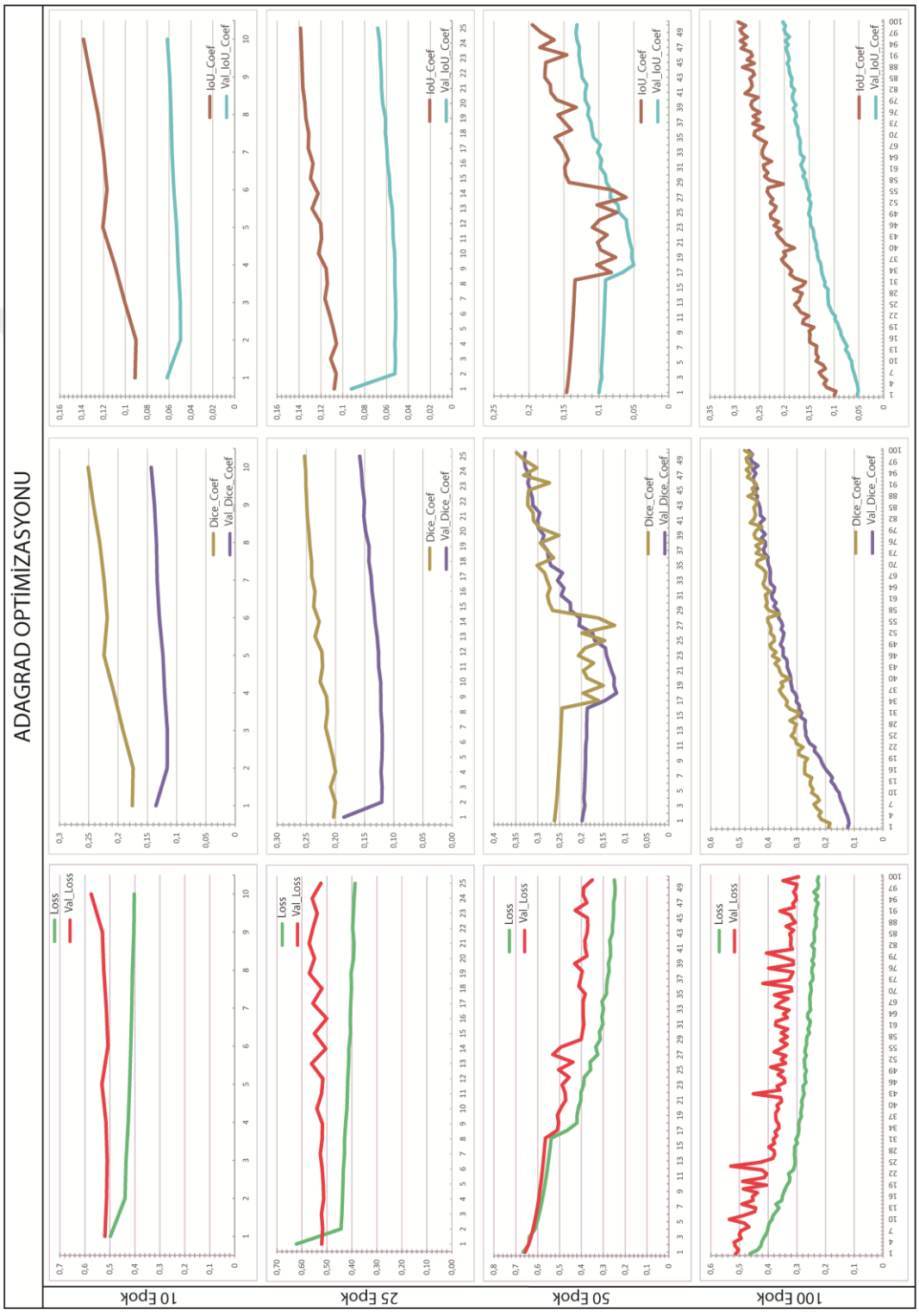
epoktaki optimizasyon grafiği incelendiğinde 48 ve 49 epoklarda artış yaşanmasının etkisi söz konusudur. 47. Epokta 0,0893 seviyelerine kadar düşmüş. 50 epokta böyle bir durum yaşanması, modelin eğitim sürecindeki donanımsal bazı zorluklar veya modelin aşırı uyum (overfitting) eğilimi göstermesinden olabilir. Eğer aşırı uyum söz konusu olmuşsa model, eğitim verilerini aşırı derecede ezberleyerek, genelleme yapma yeteneğini kaybedebilir, bu da eğitim sonuçlarını olumsuz yönde etkileyebilir. Ancak, 100 epokta kayıp fonksiyonu incelendiğin tekrar 0,4338 seviyesine düşmesi, modelin daha iyi bir performansa ulaştığını ve sistematiğe girdiğini göstermektedir. Eğitimin doğruluk değerleri Dice ve IoU ise sırasıyla 10 epokta 0,2470 ve 0,1357; 25 epokta 0,2445 ve 0,134 olarak elde edilmiştir. Bu durum, modelin eğitimi ilerledikçe hem Dice hem de IoU doğruluk değerlerinin azaldığını göstermektedir. Daha düşük doğruluk değerleri, modelin tahminlerinin gerçek etiketlerle daha az örtüştüğünü ve segmentasyon performansının düşük olduğunu işaret eder. Fakat 10 ve 25 epok için bu değerlerin yetersiz olduğunu söylemek mümkündür. Ayrıca, 50 epokta Dice doğruluk değeri 0,2799 ve IoU doğruluk değeri 0,1599'a yükselmişken, 100 epokta tekrar düşüş göstermiştir. 100 epoktaki Dice ve IoU değerler sırasıyla 0,2126 ve 0,1128 olarak bulunmuştur. Bu durum, modelin eğitimi sırasında belirli bir noktaya kadar iyileşme kaydetmesine rağmen, daha sonra performansının tekrar düştüğünü göstermektedir. Nadam optimizasyonunda modelin nesnelerin sınırlarını belirlemede biraz zorlandığı sonucunu çıkarmak mümkündür.



Şekil 4.12. Nadam için 10,25,50 ve 100 epoktaki Kayıp, Dice ve IoU grafikleri

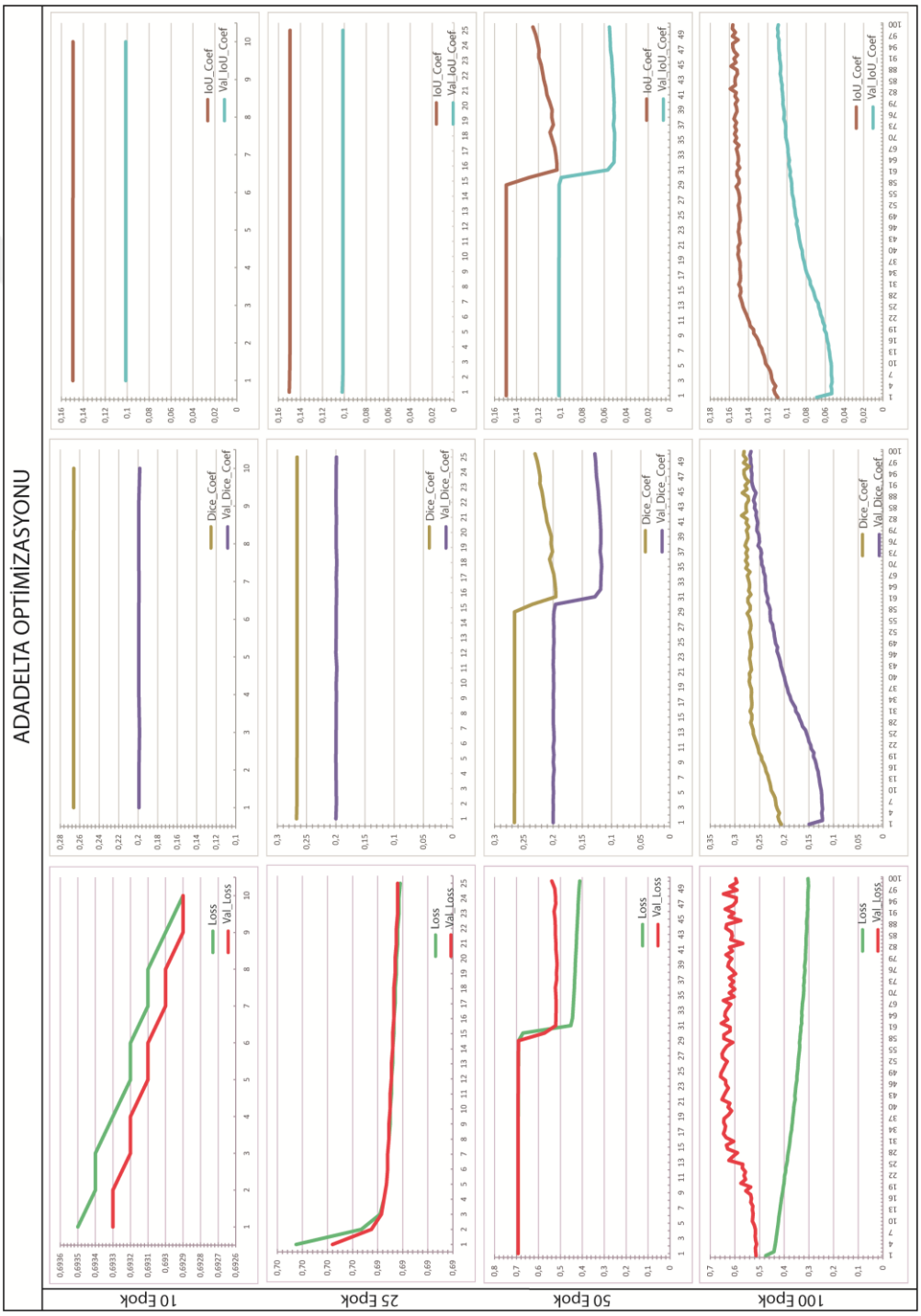
Şekil 4.13'te Adagrad optimizasyonu ait grafikler sunulmuştur. Kayıp fonksiyonu 10 epok sonunda 0,4022; 25 epokta 0,3874; 50epokta 0,2493 ve 100 epokta 0,2249 seviyelerine kadar düşmüştür. Eğitimin Dice ve IoU doğruluk değerleri ise sırasıyla 10

epokta 0,2511 ve 0,1384; 25 epokta 0,2531 ve 0,1386; 50 epokta 0,3485 ve 0,1950; 100 epokta ise 0,4820 ve 0,2933 olarak elde edilmiştir. Modelin eğitiminin sonunda oldukça düşük bir doğruluk seviyesine ulaştığını görülmektedir. Her iki metrik için doğruluk %50'nin altındadır.



Şekil 4.13. Adagrad için 10,25,50 ve 100 epoktaki Kayıp, Dice ve IoU grafikleri

Şekil 4.14'te Adadelta optimizasyonu için kayıp ve doğruluk metriklerinin grafiklerine yer verilmiştir. Kayıp fonksiyonu 10 epok sonunda 0,6929; 25 epokta 0,6922; 50epokta 0,4116 ve 100 epokta 0,3038 seviyelerine kadar düşmüştür. Eğitimin doğruluk değerleri ise sırasıyla 10 epokta 0,2661 ve 0,1495; 25 epokta 0,2667 ve 0,1499; 50 epokta 0,2308 ve 0,1251; 100 epokta ise 0,2814ve 0,1564 olarak elde edilmiştir. Bu doğruluk metriklerine bakıldığında segmentasyon işleminin çok yetersiz kaldığı görülmektedir.



Şekil 4.14. Adadelta için 10,25,50 ve 100 epoktaki Kayıp, Dice ve IoU grafikleri

Çizelge 4.9’da 6 farklı optimizasyon algoritmasının 4 farklı epok sayısında bir modelin eğitimini tamamlamak için ne kadar zamana ihtiyaç duyduğunu göstermektedir. Tüm optimizasyonlar için ilk dikkat çeken durum, epok sayısı arttıkça eğitim süresi de artmaktadır. Bu artış, algoritmalar arasında da farklılık göstermektedir. En hızlı algoritma Adam'dır, onu SGD ve Nadam takip etmektedir. En yavaş algoritma ise Adadelta'dır. Adadelta, RMSprop'un bir genişlemesi olarak kabul edilebilir ve özellikle büyük veri kümeleri veya karmaşık modellerle çalışırken etkili olabilir, ancak eğitim süresi daha uzun olduğu için tercih edilebilirlik açısından bazı dezavantajlara sahip olabilir. Eğitim süresi en düşük olan optimizasyon algoritması gibi Adagrad’dır. RMSprop ve Adam optimizasyonları, eğitim süresi bakımından diğer algoritmalarla karşılaştırılabilir bir performans sergilemektedir.

Çizelge 4.9. Optimizasyon algoritmalarının 4 farklı epok sayısında modelin eğitimini için süreleri

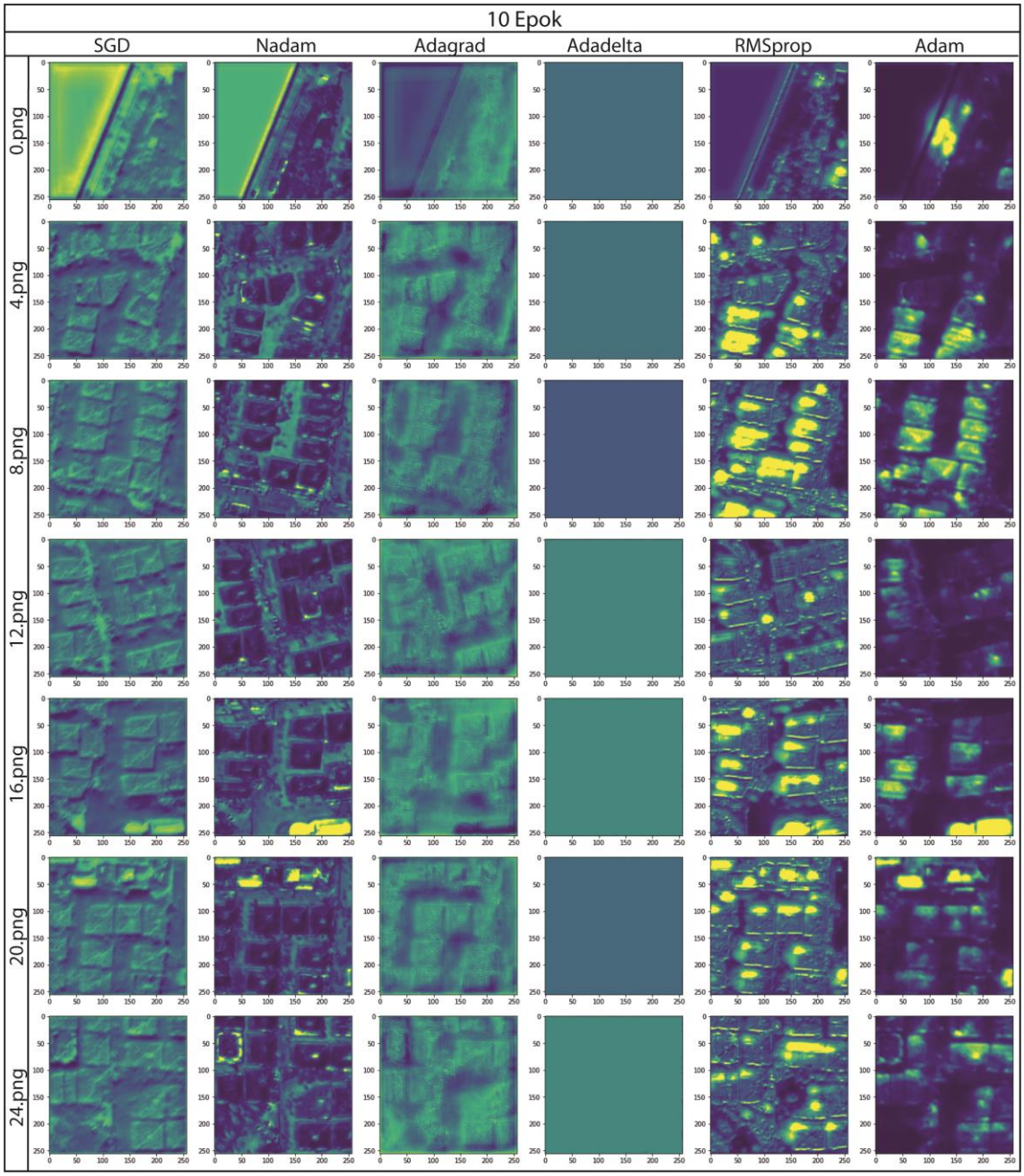
Süreler	***10 epok***	***25 epok***	***50 epok***	***100 epok***
SGD	2 sa 16 dk 50 sn	5 sa 40 dk 27 sn	11 sa 23 dk 28 sn	22 sa 34 dk 36 sn
Nadam	2 sa 18 dk 38 sn	5 sa 34 dk 8 sn	11 sa 2 dk 9 sn	23 sa 9 dk 10 sn
Adagrad	2 sa 13 dk 29 sn	5 sa 34 dk 8 sn	11 sa 6 dk 14 sn	22 sa 14 dk 46 sn
Adadelta	3 sa 4 dk 36 sn	5 sa 35 dk 49 sn	11 sa 18 dk 30 sn	30 sa 42 dk 24 sn
RMSprop	2 sa 17 dk 28 sn	5 sa 51 dk 16 sn	11 sa 15 dk 39 sn	22 sa 43 dk 23 sn
Adam	2 sa13 dk 19 sn	5 sa 28 dk 37 sn	10 sa 59 dk 21 sn	22 sa 39 dk 51 sn

Çizelge 4.10’da kullanılan optimizasyon algoritmalarının farklı metriklerde performansları incelenmiş ve değerleri sunulmuştur. Adam algoritması, tüm metriklerde en iyi sonuçları vererek bu çizelgede en iyi performansı gösteren algoritma olarak öne çıkmaktadır. Özellikle Doğruluk, Hassasiyet, F1 Skoru, Jaccard ve Dice skorlarında en yüksek değerlere ulaşmıştır. RMSprop algoritması, Adam'dan sonra en iyi performansı gösteren algoritmadır. Doğruluk, Hassasiyet, F1 Skoru ve Jaccard skorlarında Adam'a yakın sonuçlar vermiştir. Dice skoru ise Adam'dan biraz daha düşüktür. SGD ve Adagrad algoritmaları diğer algoritmalara göre daha düşük performans göstermektedir. Algoritmalara göre en iyi öğrenme oranı değişmektedir. Adam ve RMSprop için 50 ve 100 öğrenme oranları daha iyi sonuçlar vermektedir. SGD ve Adagrad için ise 25 ve 100 öğrenme oranları daha iyidir. Adadelta için her öğrenme oranında benzer sonuçlar elde edilmiştir. Bu nedenle adadelta öğrenme oranı düşük olarak görülmektedir.

Çizelge 4.10. Optimizasyonlara ait metrikler

Optimizasyon	Epok	Genel Doğruluk	Kesinlik	Duyarlılık	F1 Skoru	Jaccard	Dice
SGD	10	0,371721	0,482104	0,282094	0,380024	0,051629	0,131617
	25	0,375385	0,484417	0,288761	0,386394	0,076698	0,187839
	50	0,372576	0,482639	0,283671	0,381567	0,089803	0,193803
	100	0,372537	0,482616	0,283595	0,381482	0,073086	0,183837
Nadam	10	0,375034	0,584184	0,288175	0,385917	0,094153	0,193847
	25	0,474402	0,583784	0,287034	0,384843	0,080388	0,206454
	50	0,479940	0,584513	0,217866	0,666609	0,229015	0,464190
	100	0,483091	0,589290	0,302749	0,399718	0,178393	0,373334
Adagrad	10	0,471504	0,581968	0,281693	0,379631	0,057190	0,128649
	25	0,472505	0,582595	0,283541	0,381440	0,052340	0,136614
	50	0,490200	0,5937	0,315640	0,411922	0,091732	0,224135
	100	0,472882	0,582831	0,284235	0,382116	0,140925	0,327683
Adadelta	10	0,471095	0,581712	0,280938	0,378890	0,100798	0,196760
	25	0,472424	0,582544	0,283392	0,381294	0,104651	0,194210
	50	0,471525	0,581981	0,281733	0,379670	0,085303	0,163786
	100	0,471530	0,581984	0,281741	0,379678	0,082857	0,201895
RMSprop	10	0,4839	0,589798	0,304350	0,401345	0,078393	0,170598
	25	0,509743	0,606325	0,350251	0,443614	0,158567	0,382303
	50	0,692557	0,661947	0,486833	0,560055	0,509046	0,557995
	100	0,787611	0,758193	0,580196	0,655079	0,708474	0,850842
Adam	10	0,486314	0,591393	0,308304	0,404640	0,118396	0,28093
	25	0,564452	0,642774	0,441602	0,522246	0,208807	0,48359
	50	0,645208	0,626379	0,499962	0,586386	0,429015	0,72215
	100	0,822340	0,754763	0,571154	0,647338	0,818911	0,91568

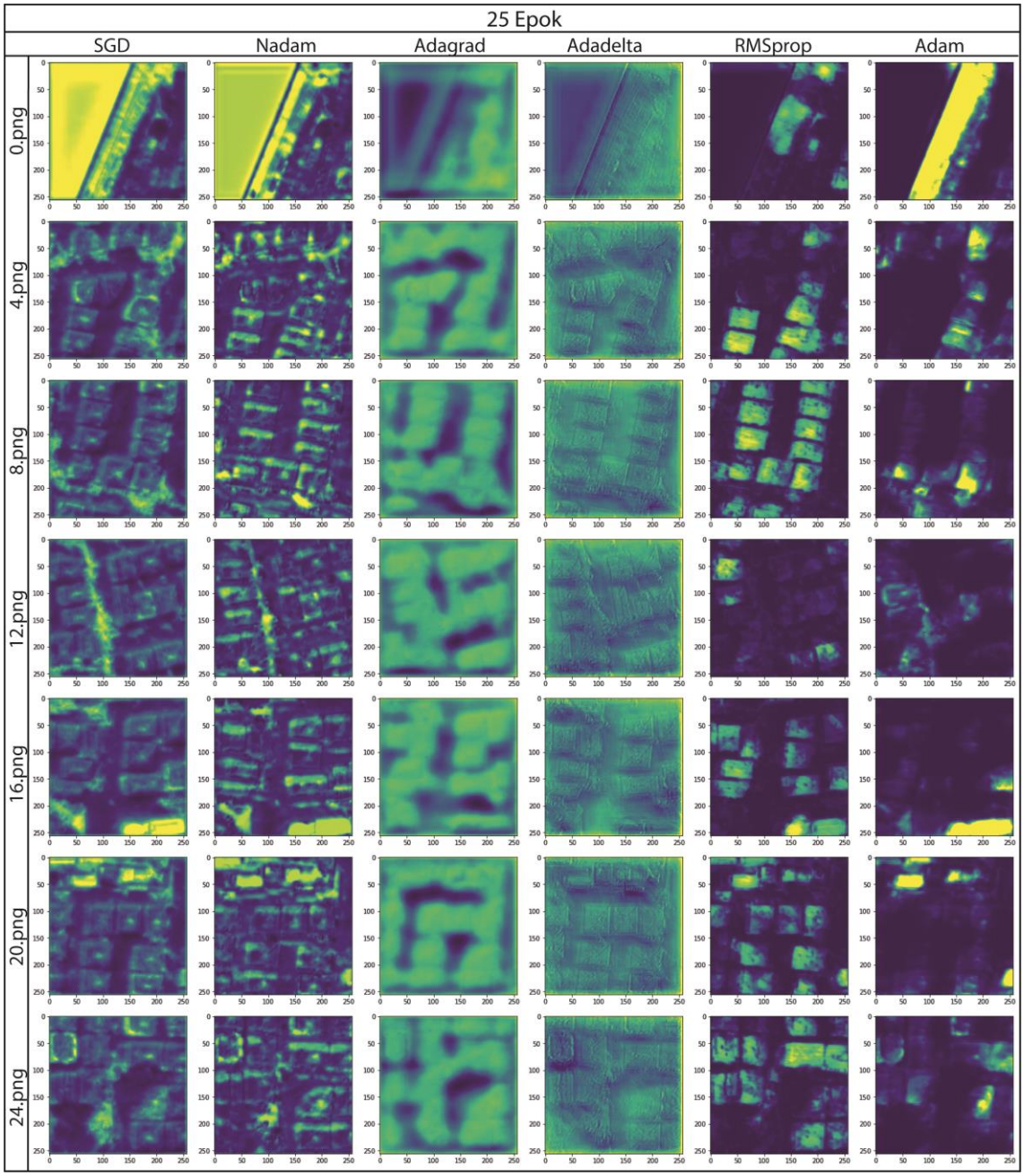
SGD, Nadam, Adagrad, Adadelta, RMSprop, Adam optimizasyon algoritmaları kullanılarak ve her algoritma için, farklı epoklardaki 7 test görüntüsünün sonuçlarına ait görselleri verilmiştir. Şekil4.15'te 10 epok, Şekil4.16'da 25 epok, Şekil4.17'de 50 epok ve Şekil4.18'de 100 epok için görseller sunulmuştur. 10 epoktaki görseller incelendiğinde, öğrenme oranının başladığı görülmekte fakat anlamlı sonuçlar üretmediği anlaşılmaktadır. Özellikle SGD, Adagrad ve Adadelta optimizasyonunda şekiller neredeyse tek renk olarak gözükmektedir. Diğer üç optimizasyonda ise, bina çıkarımı için bilginin var ama yetersiz olduğu anlaşılmaktadır. RMSprop ve Adam optimizasyonları kıyasla daha iyi sonuç vermiştir.



Şekil 4.15. 10 epok sonucundaki tahmin görüntüleri

Şekil4.16'da 25 epok için çıktı sonuçları yer almaktadır. Bu epoktaki görseller incelendiğinde, SGD, Adagrad ve Adadelta optimizasyonunda öğrenmenin çok yavaş gerçekleştiği görülmektedir. SGD diğer iki optimizasyona göre daha iyi olarak gözükmektedir. Nadam optimizasyonunda 16 numaralı görüntüde sadece birkaç bina tespit edebilmiş, diğer görüntülerde öğrenme işleminin var olduğu fakat yetersiz olduğu görülmektedir. Adam optimizasyonu içinde benzer bir durum söz konusudur. 25 epokta

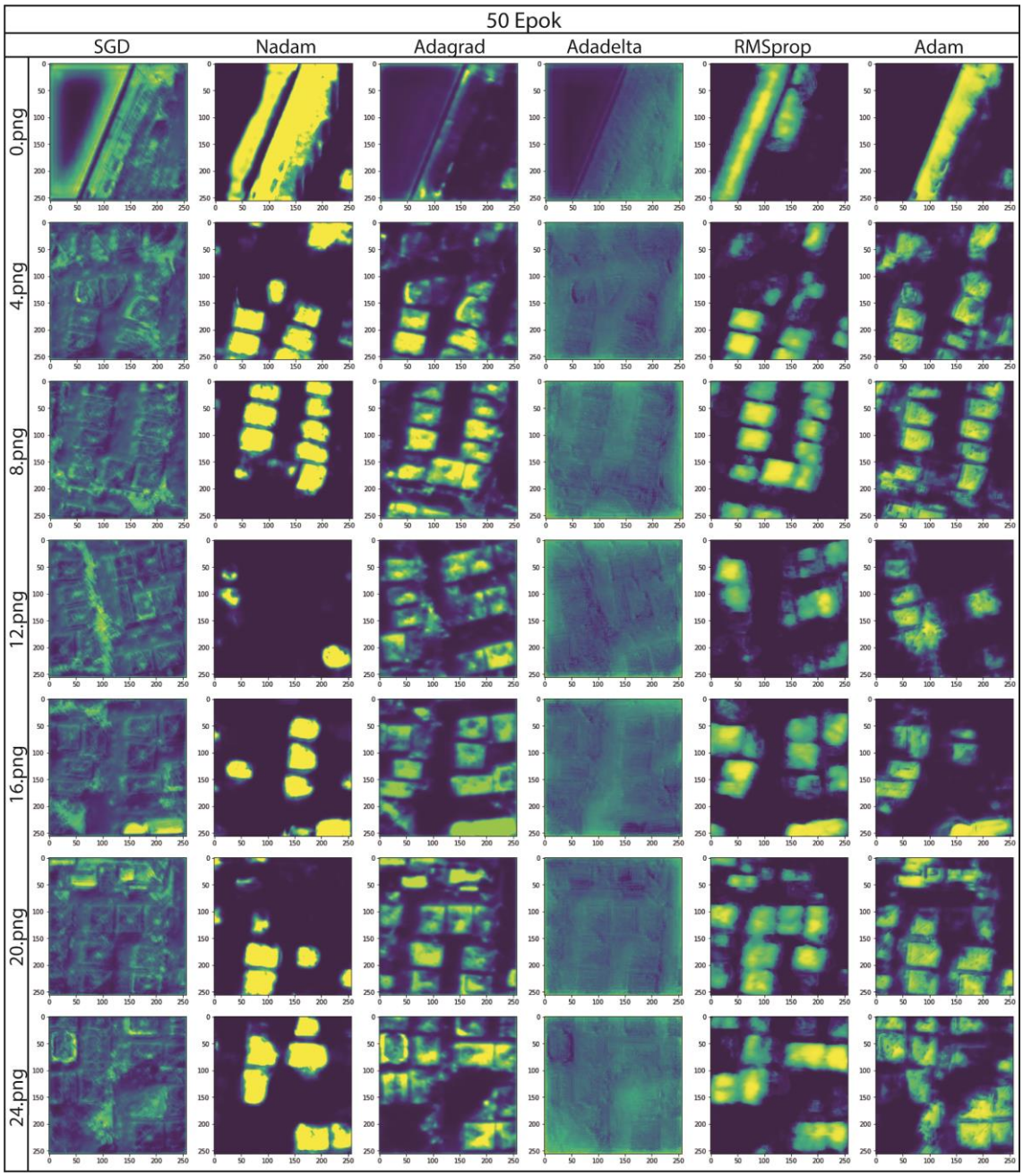
en iyi sonuç RMSprop vermiştir. Çoğu bina konumları belirsiz de olsa tespit edilmeye başlandığı görülmektedir.



Şekil 4.16. 25 epok sonucundaki tahmin görüntüleri

Şekil 4.17'de 50 epok için çıktı sonuçları yer almaktadır. SGD ve Adagrad öğrenme durumlarının Adadeltaya göre daha iyi olduğu görülmektedir. Bu uygulamada Adadelta'nın, diğer optimizasyonlara göre öğrenme oranı en yavaş optimizasyon olduğu tespit edilmiştir. Çünkü 50 epokta dahi, bina kenarları net olarak tespit edilememiştir.

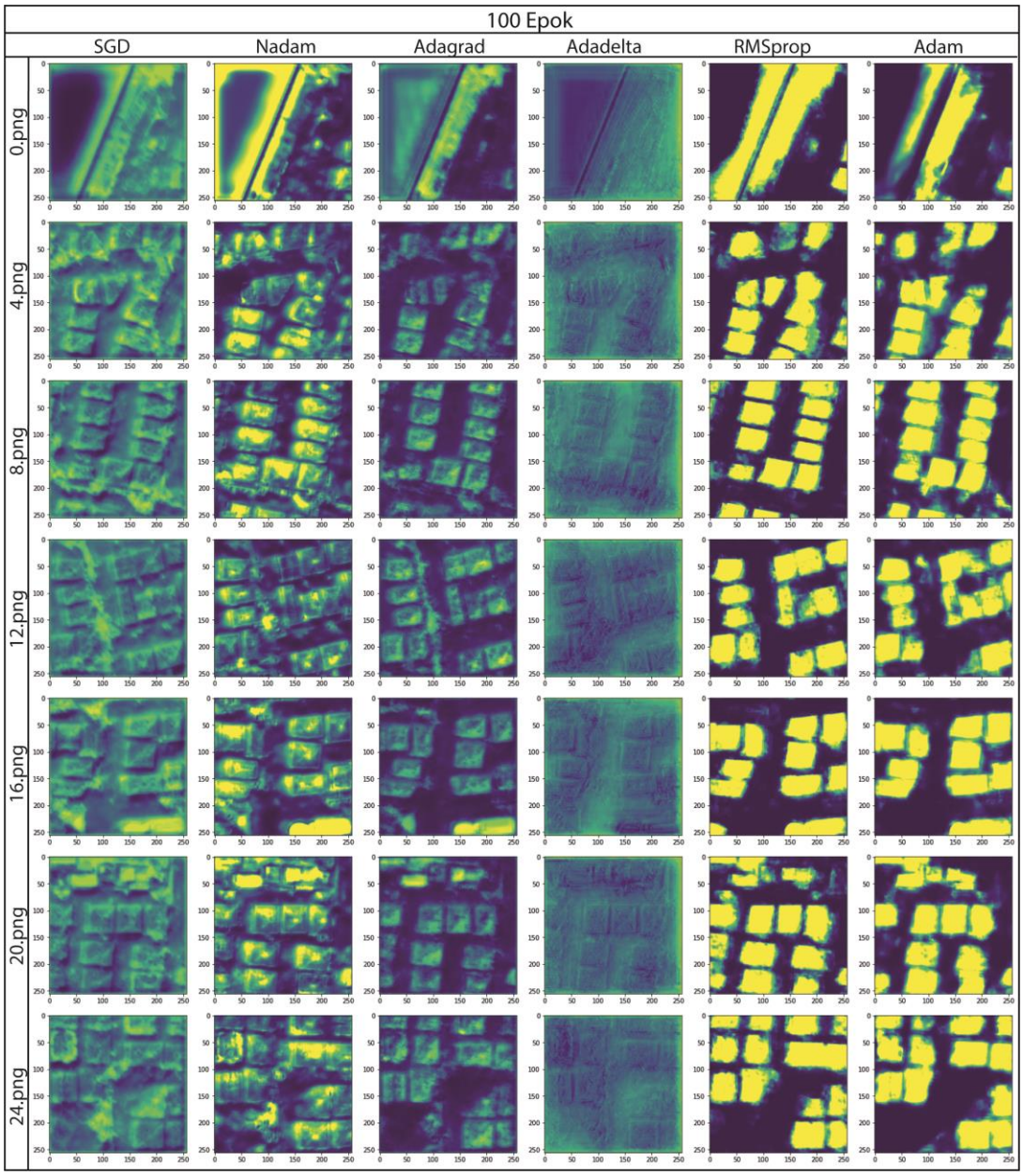
Nadam optimizasyonunda tespit edilen binalar daha net olarak gözükmektedir. Fakat hala tespit edilemeyen binaların olduğu görülmektedir. RMSprop ve Adam optimizasyonunda bina konumları kısmen tespit edildiği ama hala yetersi olduğu görülmektedir.



Şekil 4.17. 50 epok sonucundaki tahmin görüntüleri

Şekil 4.18'de 100 epok için çıktı sonuçları yer almaktadır. Adadelta optimizasyonu 100 epokta dahi yeterli öğrenme seviyesine erişememiştir. SGD ve Adagrad optimizasyonu öğrenmeye başlamıştır fakat anlamlı sonuç üretmek için yeterli değildir. Adagrad tam olarak binayı ifade edemese de maskelenmiş görüntüyle bina

konumları eş şekilde olduğu görülmektedir. En iyi sonuç diğer epoklarda olduğu gibi Adam ve RMSprop optimizasyonundan elde edilmiştir. Doğruluk anlamında metriksel olarak bakıldığında Adam optimizasyonunun %3 oranında RMSprop'tan daha iyi olduğu ifade edilebilir.



Şekil 4.18. 100 epok sonucundaki tahmin görüntüleri

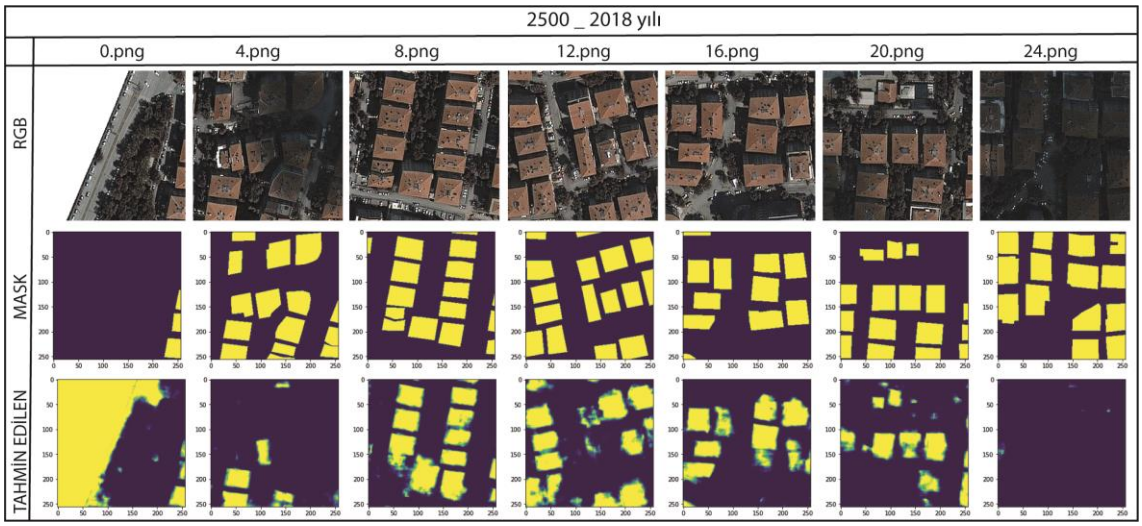
Epok sayısının artmasıyla tüm görüntülerdeki öğrenme oranının daha iyi olduğu söylenebilmektedir. Fakat Adadelta'nın öğrenme süreci yavaş gerçekleştiği için, binalar tespit edilememiştir. SGD, Nadam, Adagrad, RMSprop ve Adam'da öğrenme sürecinin 0 epoğa göre daha iyi olduğu söylenmesi mümkündür.

0 numaralı görüntüde, görüntünün yarısının boş olmasından kaynaklı tüm optimizasyon yöntemlerinde binalar tam olarak tespit edilememiştir. Görüntünün olan kısmı ve olmayan kısmını tam algılanamamış ve büyük oranda yanlış segmentasyon işlemi gerçekleştirilmiştir. 0 numaralı görüntünde bulunan 3 adet binadan RMSprop optimizasyonu ile 2 tanesi ve Adam optimizasyonu ile 1 tanesi tespit edilmesi mümkün olmuştur. Diğer optimizasyonlarda bu görüntü için işlem başarısız olmuştur. 4 numaralı görüntünün sağ üst kısmının gölgede kalmasından ve bina şekillerinin geometrik yapıda olmamasından dolayı karışık dursa dahi, anlamlı sonuçlar ürettiği görülmektedir. Özellikle gölgede 2 adet binanın tespit edilmesi mümkün olduğu görülmüştür. 8,16 ve 20 numaralı görüntülerde Adam ve RMSprop optimizasyonları sayesine tüm binalar tamamen bulunmuştur. 12 numaralı görüntüde RMSprop'ta tüm binalar tespit edilirken, Adam optimizasyonunda sağ alt köşedeki bir binanın tespit edilemediği görülmektedir. 24 numaralı görüntüde sağ alt köşede bulunan üçgen şeklindeki bina dışındaki tüm binalar tespit edilmiştir. Benzer şekilde bu bina da gölge etkisinde olmasına rağmen iyi sonuç verdiği görülmüştür.

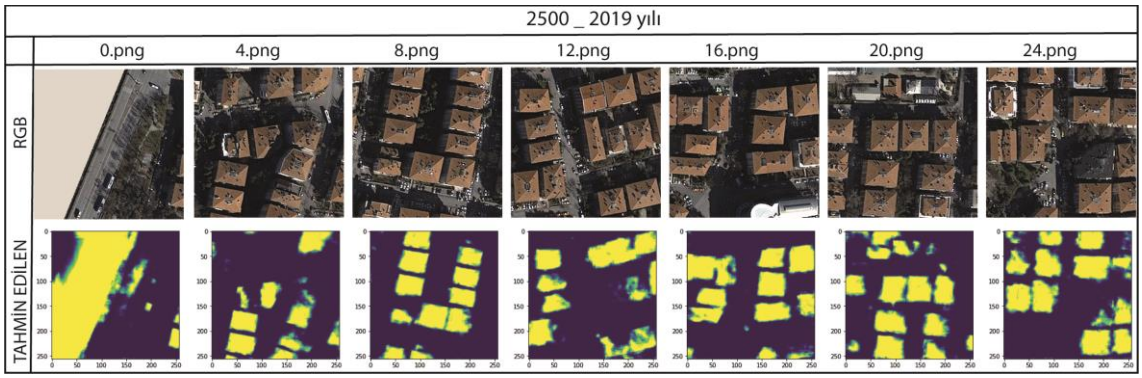
4.4. Bina için Farklı Eğitim Sayılarının Sonuç Değerlendirmesi

Adam optimizasyonu, tüm metriklerde en iyi sonuçları ve en iyi performansı gösteren optimizasyon olduğundan daha fazla veri ile eğitilmesi durumundaki sonuçlar incelenmek istenmiştir. Bu nedenle eğitim sayısı 1000 iken bu sayı arttırılmış ve iki farklı eğitim seti oluşturulmuştur. Yeni oluşturulan veri setlerindeki eğitim sayılarının boyutları 2500 ve 5000 görüntüden oluşturulmuştur. Ayrıca 2018, 2019,2020 ve 2021 yıllarındaki bina tespitleri, görüntülerdeki farklılıklardan kaynaklı durumların sonuca etkisi araştırılmıştır. 2020 yılındaki görüntü 25 Aralık tarihine aitken, 2021 yılındaki görüntü 23 Ocak tarihine aittir. Her iki görüntü de kış mevsiminde toplanmıştır. Çatılar üzerindeki beyazlığın kar olduğu açıkça söylenebilmektedir. İlk eğitim seti 2500 olarak seçilmiştir. Şekil 4.19'da 2018 yılı için tahmin görüntüsü verilmiştir. 0 numaralı görüntüde 3 adet tahmin etmesi gereken binayı tespit etmiştir. Fakat görüntünün içeriğinin bir kısmından kaynaklı sol taraftaki kısmı da bina olarak belirlemiştir. 4 numaralı görüntüde gölge faktörünün olması sebebiyle çoğu yapı tespit edilememiştir. 8 numaralı görüntüde sol alt tarafta bulunan iki bina dışındaki diğer binaların tespiti gerçekleştiği görülmektedir. 12,16 ve 20 numaralı görüntüde büyük oranda binaların tespit edilmiştir. 24 numaralı görüntünün tamamen gölgede olması sonucu negatif olarak etkilemiştir. Neredeyse hiçbir

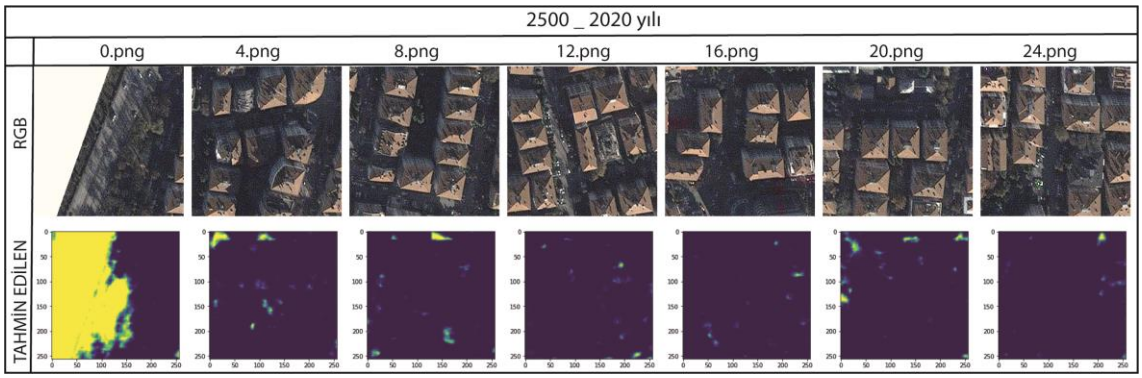
bina tespiti gerçekleştirilememiştir. Şekil 4.20'de 2019 yılına ait tahmin edilen görüntüler sunulmuştur. Burada 0, 8 ve 16 numaralı görüntülerde tüm binalar tespit edilirken, 4, 12, 20 ve 24 numaralı görüntülerde birkaç binanın eksik olduğu tespit edilmiştir. Şekil 4.21'de 2020 yılına ait tahmin edilen görüntüler sunulmuştur. Elde edilen sonuç doğrultusunda havanın kış mevsiminden ötürü kar yağışlı olması görüntü üzerinde gürültü olmasına sebebiyet vermiştir. Bu durumda görüntü üzerinden öğrenme durumunu etkilemiş ve doğru bir tahmin yapılamamıştır. Benzer durum Şekil 4.22'de verilen 2021 yılı içinde geçerli olduğu görülmektedir. Fakat burada kar sadece çatılarda gözükmektedir.



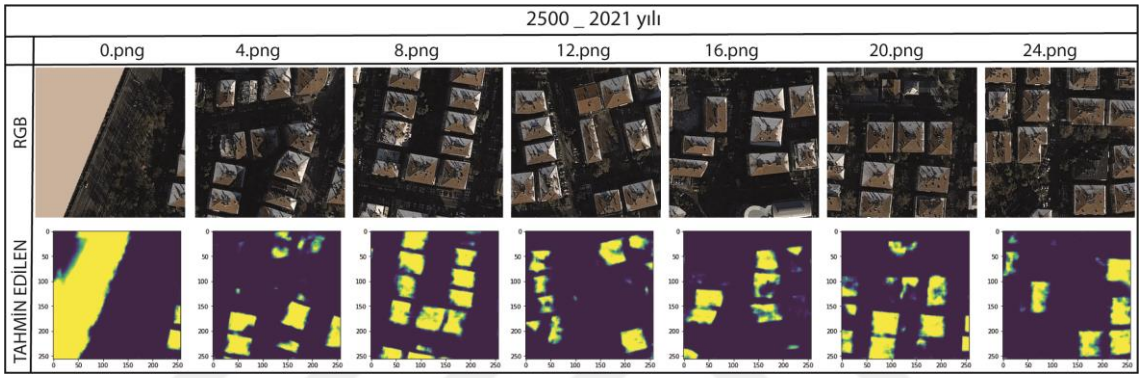
Şekil 4.19. Eğitim seti 2500 iken 2018 yılının tahmin görüntüleri



Şekil 4.20. Eğitim seti 2500 iken 2019 yılının tahmin görüntüleri

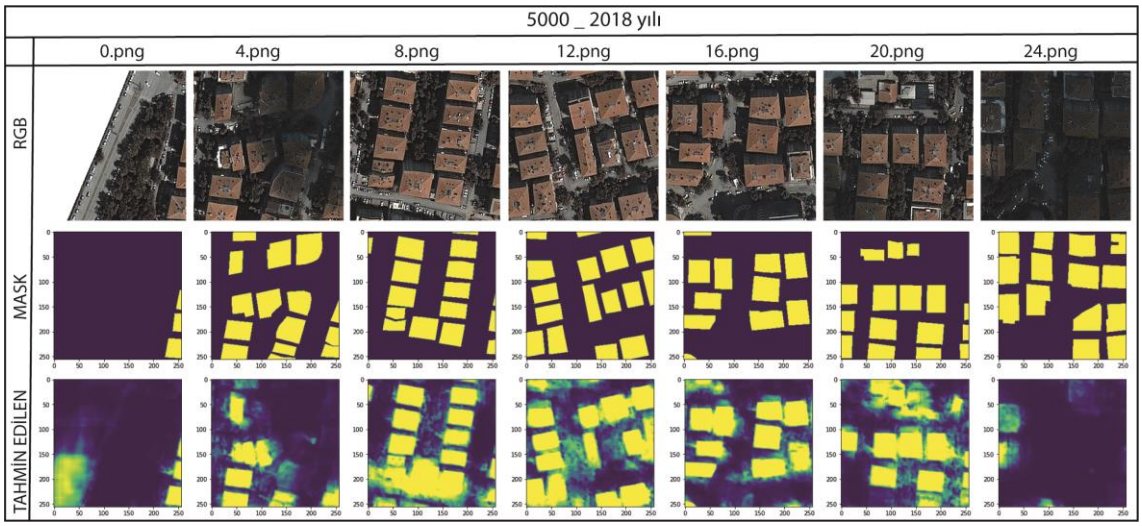


Şekil 4.21. Eğitim seti 2500 iken 2020 yılının tahmin görüntüleri

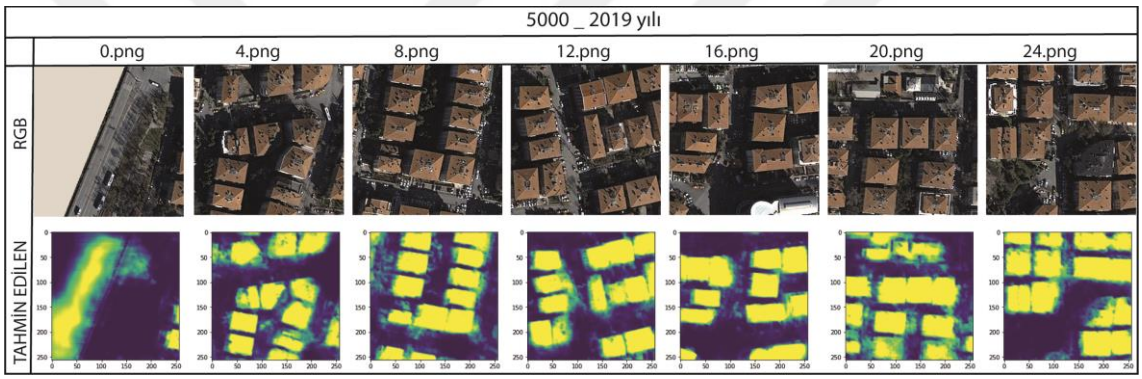


Şekil 4.22. Eğitim seti 2500 iken 2021 yılının tahmin görüntüleri

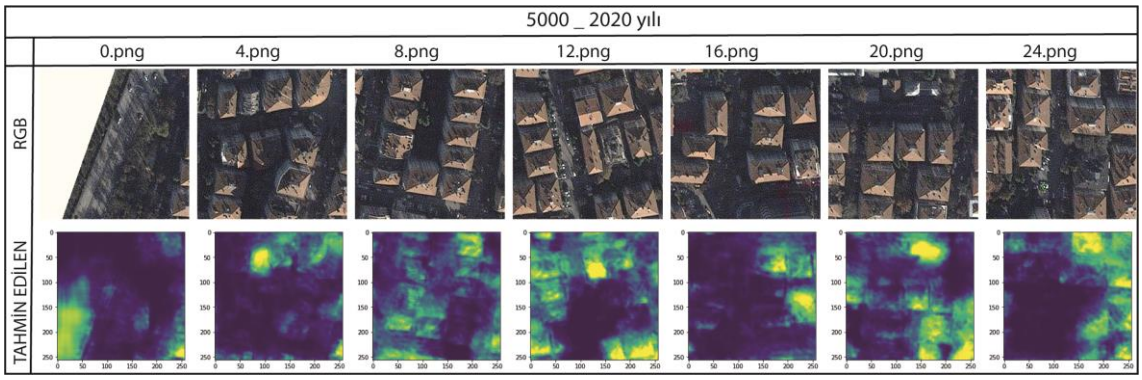
2500 eğitim verisi ile 1000 eğitim verisi neredeyse benzer sonuçlar ürettiği görülmektedir. Bu nedenle eğitim sayısı 5000 seçilerek aynı işlemler uygulanmıştır. Elde edilen bulgu sonuçları Şekil 4.23, Şekil 4.24, Şekil 4.25, ve Şekil 4.26'de sunulmuştur. En iyi sonucun 2019 yılında elde edildiği görülmüştür. Fotoğraf baz alınarak incelendiğinde, 0 numaralı görüntünün 2018 yılında daha iyi sonuç verdiği açıkça anlaşılmaktadır. Geriye kalan 6 adet görüntünün hepsi 2019 yılında elde edilen görüntüden olduğu tespit edilmiştir. 2020 yılı için tahmin işleminin başarısız olduğu tespit edilmiştir. Fakat 2021 yılında karın sadece çatılarda olması bina çıkarımın yapılmasının mümkün olduğunu göstermektedir. 12 numaralı fotoğrafta orta kısımda bulunan dikdörtgen şeklinde bina hariç, tüm görüntülerdeki binaların çıkarım işleminin doğru olduğu görülmektedir.



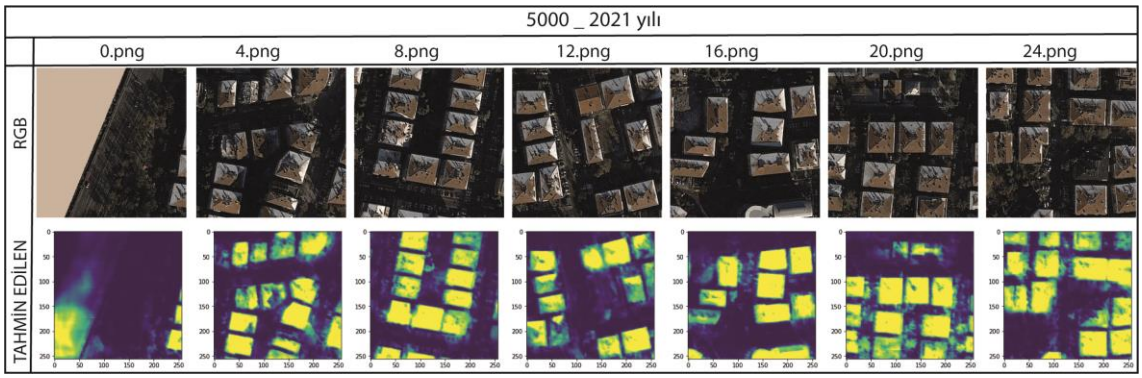
Şekil 4.23. Eğitim seti 5000 iken 2018 yılının tahmin görüntüleri



Şekil 4.24. Eğitim seti 5000 iken 2019 yılının tahmin görüntüleri



Şekil 4.25. Eğitim seti 5000 iken 2020 yılının tahmin görüntüleri



Şekil 4.26. Eğitim seti 5000 iken 2021 yılının tahmin görüntüleri

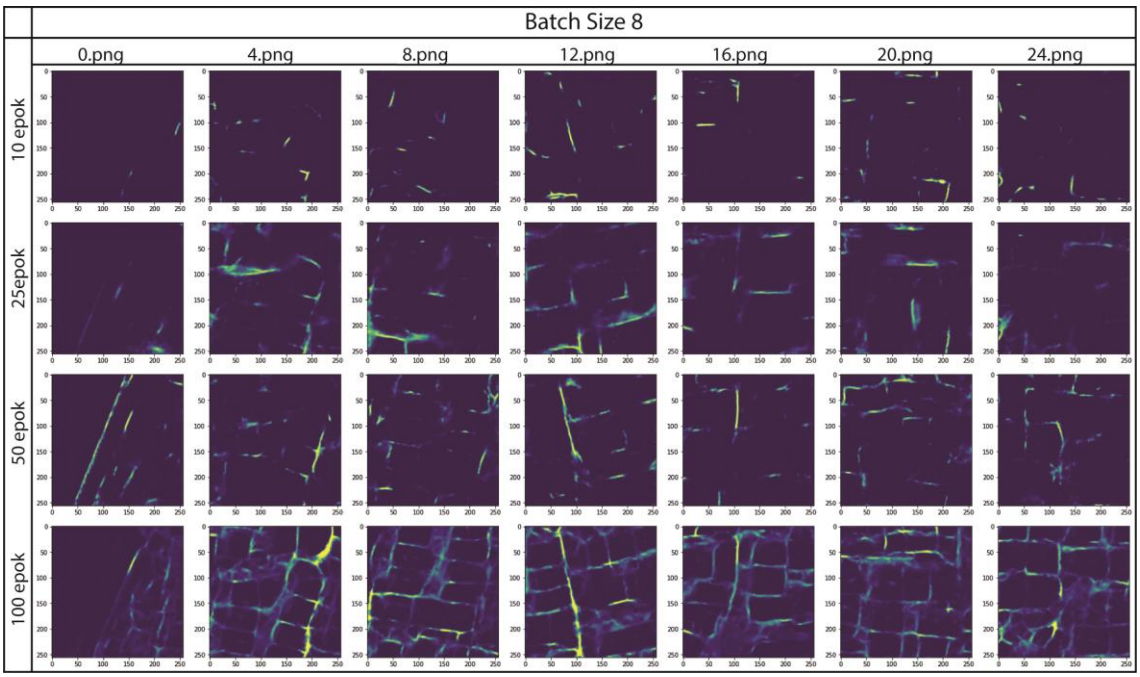
Bu kısımda yapılacak temel çıkarım ise, görüntü üzerinden tahmin işlemi yapılacaksa eğer, görüntülerin uygun mevsimde toplanmış olması önem arz etmektedir. Çünkü aynı bölgeye ait 4 farklı yılın görüntüsünden çeşitli tahmin sonuçları elde edilmiştir. Yaz mevsiminde toplanmış olan 2019 yılına ait görüntülerin tahmin sonuçları oldukça başarılıdır. Fakat, 2020 yılında elde edilen görüntüden neredeyse hiç tahmin yapılamamıştır. 2021 yılı da kış mevsimine ait bir görüntüdür. Bu yılda, 2500 eğitim verisindeki sonuçları kısmen kötü olsa dahi, 5000 eğitim verisinden elde edilen sonuçlarda aynı şeyi söylemek mümkün olmaz. Çünkü 5000 eğitim setinde verinin öğrenme durumu söz konusudur. Eğitim sayısındaki artış görüntülerde iyileştirmeler sağlamıştır.

4.5. Yol için Farklı Batch-size'ın Sonuç Değerlendirmesi

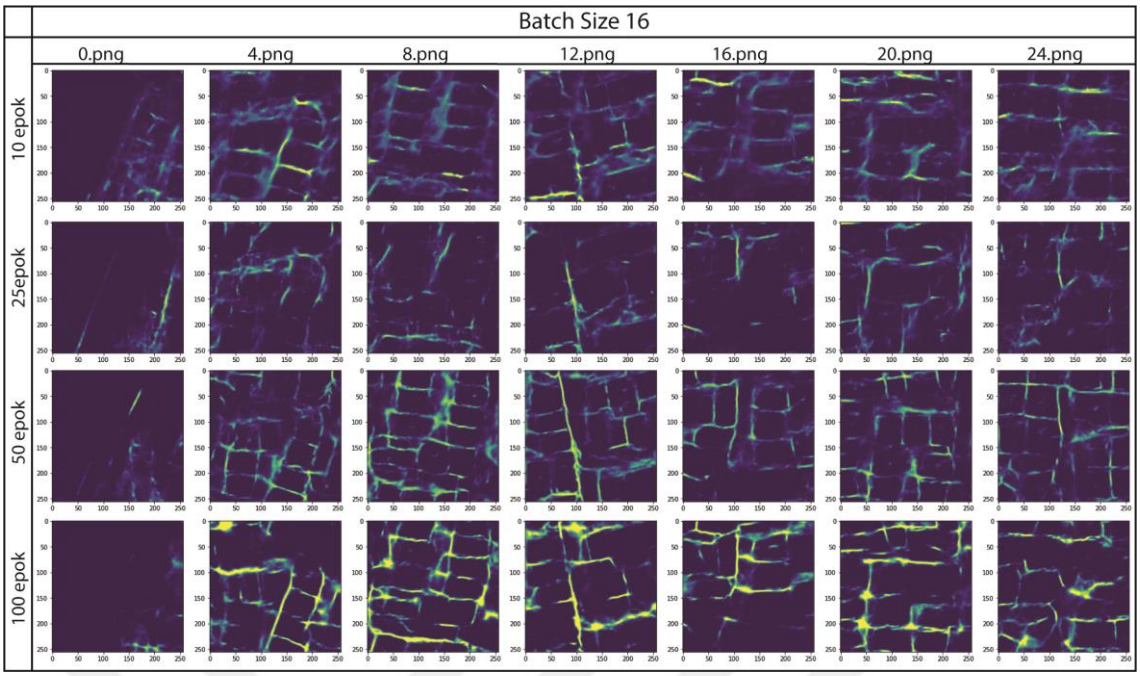
Görüntü segmentasyonunda sıkça gerçekleştirilen işlemlerden biri de yol ağlarının belirlenmesidir. Binaların segmentasyon işleminde kullanılan U-net mimari bu kısımda tekrar uygulanacaktır. Mimarinin temelinde, kodlama aşamasında, görüntüden özellik haritaları çıkarılırken, kod çözme aşamasında ise bu özellik haritaları tekrar girdi boyutuna dönüştürülerek segmentasyon gerçekleştirilir. Tüm ağ eğitimi sırasında giriş ağının görüntü boyutu 256×256'dır. Bu çalışmadaki model eğitimi için 10 epok, 25 epok, 50 epok ve 100 epok olacak şekilde 4 farklı epok durumu kullanıldı. Batch boyutu için ise 3 farklı durum seçilmiştir. Batch boyutu, eğitim sürecinde aynı anda işlenen örneklerin sayısını belirler ve derin öğrenme modelinin öğrenme sürecini etkiler. Bu bağlamda, U-net mimarisi kullanılarak gerçekleştirilen bir çalışmada batch boyutlarının (8, 16 ve 32) segmentasyon performansına etkisi incelenmiştir. Bunun için Adam optimizasyonu ve öğrenme oranı 1e-3 sabit olarak seçilmiştir. Binalarda batch boyutu 16 daha iyi sonuç

vermişti. Benzer durumun burada söz konusu olup olmayacağını incelemektir. Çalışmanın amacı, farklı batch boyutlarının yol ağı tabanlı segmentasyon performansını nasıl etkilediğini göstermektedir. Şekil4.27'de Batch boyutu 8'e ait tahmin edilen görüntüler bulunmaktadır. Oluşturulan görüntülerin çıktı sonuçları her defasında mor renge yakın olduğu görülmektedir. Bu durum da modelin eğitimi gerçekleştirilemediği kanısına varılmaktadır. Batch boyutunun 8 seçilmesi durumunda yol ağlarının belirlenmesinde yetersiz kaldığı görülmektedir. Şekil4.28'de Batch boyutu 16 ve Şekil4.29'da Batch boyutu 32'de elde edilen tahmin görüntüleri verilmiştir. 10,25,50 epoktaki sonuçlar anlamlı olmasa da 100 epoktaki sonuçlar incelendiğinde yol ağlarının tespit edildiği görülmektedir. En iyi sonucun ise batch boyutu 16'da verdiği tespit edilmiştir.

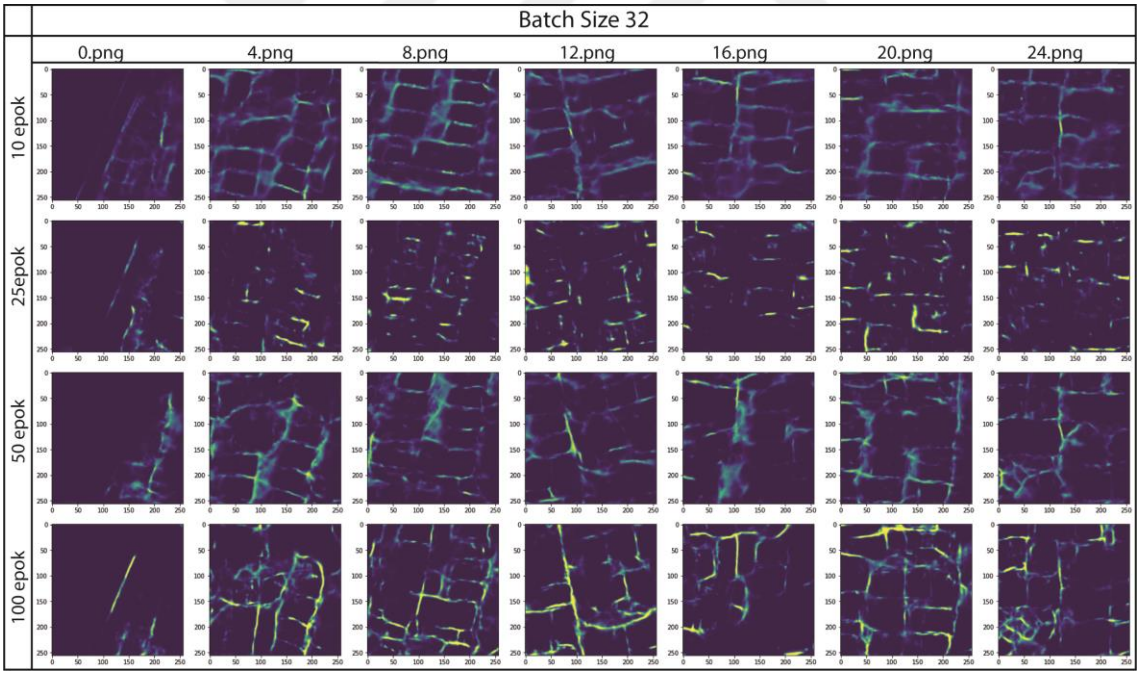
Tüm görüntüler için metrik değerler hesaplanmamıştır. Çalışmada yol ağlarının maskelenmiş görüntüleri oluşturulmuştur. Fakat temelde ana yollar tanımlanmıştır. Tahmin edilen görüntülerde tüm yolları öğrenmeye çalıştığı görülmektedir. Metrik değerlerin düşük gelmesinden dolayı tahmin edilen görüntüler sunulmuştur. Modelin eğitim performansının oldukça yüksek olduğu kayıp, dice ve IoU grafiklerinden okunmaktadır.



Şekil 4.27. Yol ağı batch boyutu 8 için elde edilen tahmin görüntüleri



Şekil 4.28. Yol ağı batch boyutu 16 için elde edilen tahmin görüntüleri



Şekil 4.29. Yol ağı batch boyutu 32 için elde edilen tahmin görüntüleri

Çizelge 4.11'de yol ağlarının farklı batch boyutlarındaki metrik değerleri ifade edilmiştir. En yüksek doğruluğun %86.14 oranıyla BS_16 ve 100 epokta olduğu görülmektedir. En düşük doğruluk değeri ise, BS_8 10 epokta olduğu görülmektedir. Tüm batch boyutlarında epok sayıları artıkça doğruluk değerlerinin arttığı tespit edilmiştir.

BS_32 de 25 epoktan sonra doğruluk metriğinde bir düşme ve sonrasında artış olmuştur. Tahmin edilen ve gerçek görüntü arasındaki benzerliği ölçmek için dice ve jacard metriklerinden yararlanılmıştır. En yüksek jaccard değeri %72.88 ve en yüksek dice değeri %84.79 olarak bulunmuştur. BS_16 da oldukça iyi sonuçlar verdiği görülmektedir. Genel olarak ifade edilirse, doğruluk, duyarlılık, F1 skoru, Jaccard ve Dice katsayıları da epok boyutu ile artma eğilimindedir.

Çizelge 4.11. Yol ağı batch size metrik değerleri

Batch size		Genel Doğruluk	Hassasiyet	Duyarlılık	F1 Skoru	Jaccard	Dice
BS_8	10	0,583868	0,676534	0,520893	0,588251	0,208872	0,352172
	25	0,705371	0,766628	0,694277	0,728516	0,339243	0,600825
	50	0,764278	0,789381	0,732273	0,759680	0,409263	0,695980
	100	0,806008	0,823580	0,784419	0,803453	0,419420	0,720454
BS_16	10	0,597693	0,665369	0,495360	0,567247	0,248727	0,411864
	25	0,657751	0,708205	0,585856	0,640787	0,468777	0,655633
	50	0,791002	0,732732	0,633592	0,679309	0,578887	0,735288
	100	0,861456	0,820826	0,801453	0,805262	0,728829	0,847949
BS_32	10	0,601874	0,668163	0,502277	0,573067	0,218681	0,375850
	25	0,732143	0,764253	0,689657	0,724827	0,289130	0,586766
	50	0,673554	0,719779	0,608770	0,659251	0,348783	0,658056
	100	0,691887	0,733364	0,634987	0,680330	0,399022	0,714903

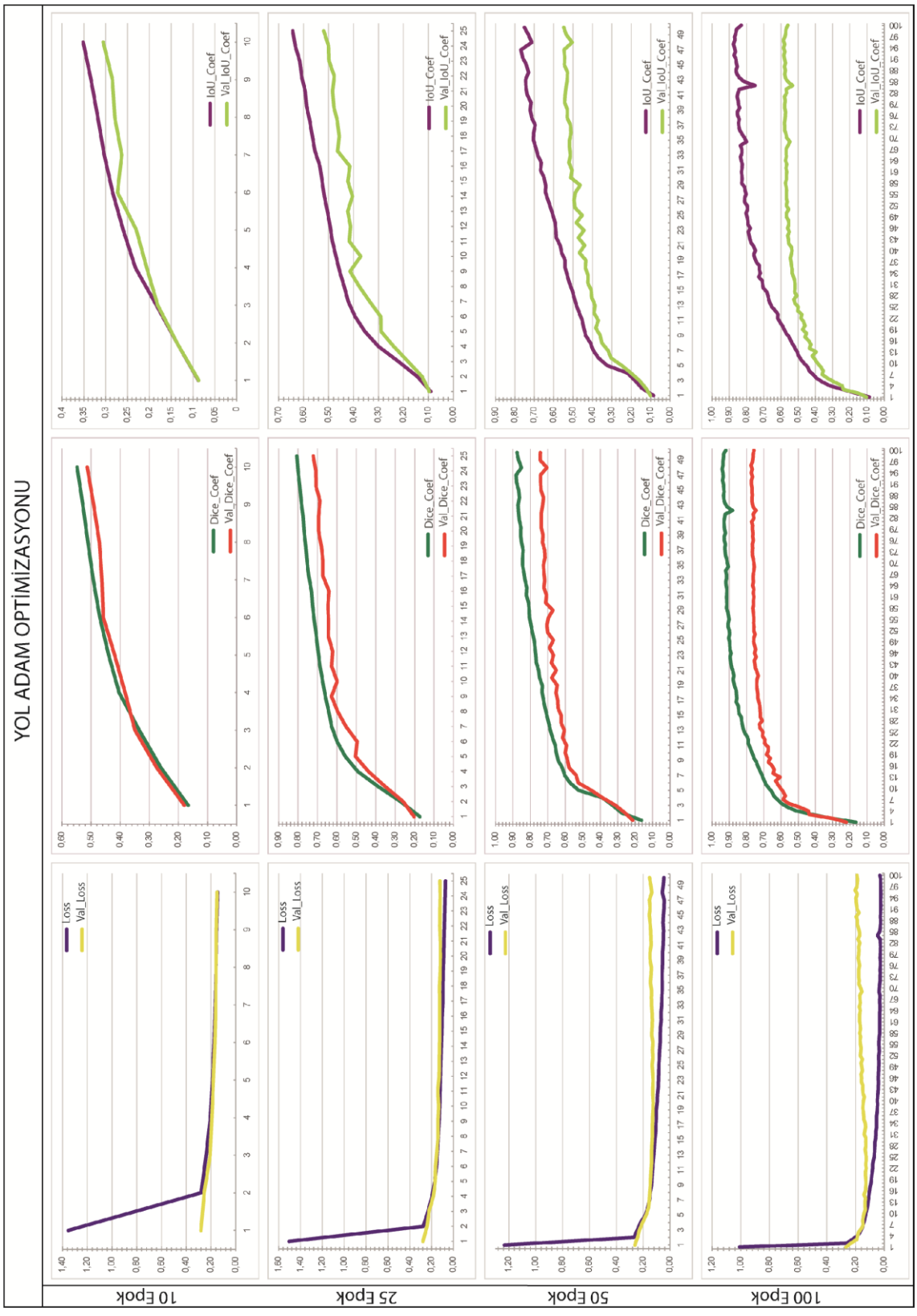
4.6. Yol için Farklı Optimizasyon Yöntemlerinin Sonuç Değerlendirmesi

Bu tez çalışmasının devamında, yolların segmentasyonu için U-Net mimarisini ve iki adet optimizasyon yöntemi kullanılmıştır. Bu optimizasyonlar Adam ve RMSproptur. İki optimizasyonun seçilmesinin temel nedeni, bina segmentasyonunda en iyi sonucu veren yöntemdir. Kentsel alanlardaki binaların, yolların ya da çeşitli sınıfların çıkarılmasında U-net mimarisi daha çok tercih edilmektedir (Hao ve ark., 2023). Şekil4.30'da Adam ve Şekil4.33'te RMSprop optimizasyonlarına ait kayıp ve doğruluk (Dice ve IoU) grafikleri sunulmuştur. Her bir parametrenin 10epok, 25epok, 50epok ve 100 epoktaki durumları incelenmiştir. Ayrıca Çizelge 4.31'de Adam optimizasyonundan ve Çizelge 4.32'de RMSprop optimizasyonu sonucu elde edilen tahmin görüntüleri bulunmaktadır.

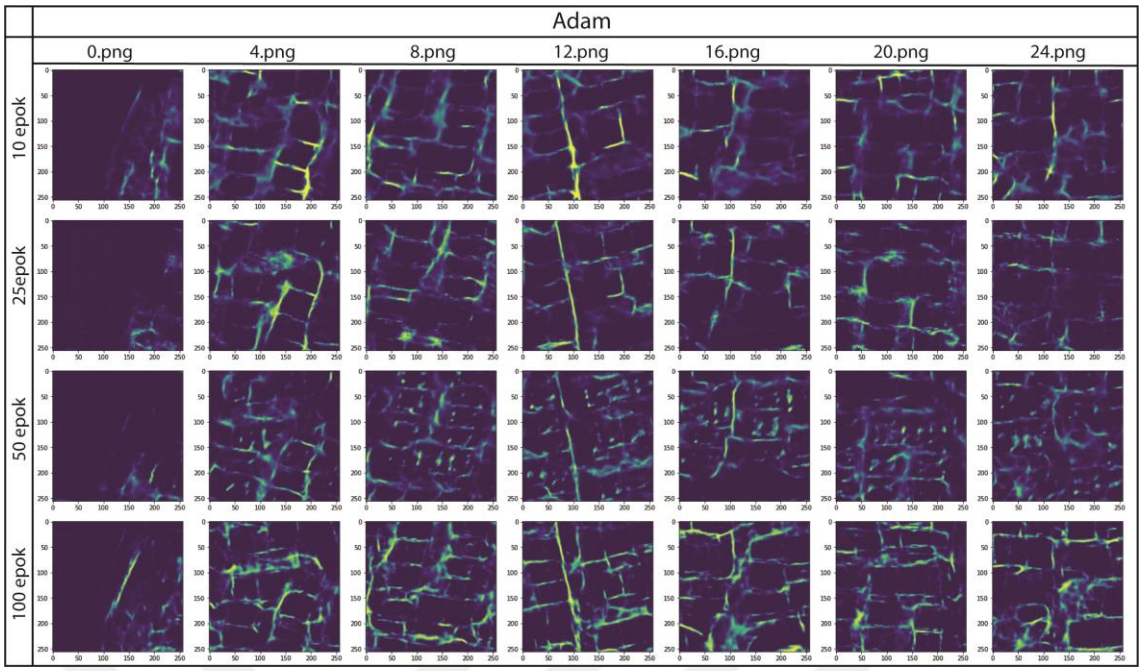
Şekil 4.30’de Adam optimizasyonu için kayıp ve doğruluk metriklerinin grafiklerine yer verilmiştir. Kayıp fonksiyonu 10 epok sonunda 0,1461; 25 epokta 0,0731; 50epokta 0,0465 ve 100 epokta 0,0305 seviyelerine kadar düşmüştür. Kayıp başlangıçta 0,1461 iken, 100 epok sonunda 0,0305 seviyelerine kadar düşmüştür. Başlangıçta yüksek olan kayıp, model eğitildikçe kademeli olarak düştüğü görülmektedir. Bu, modelin eğitim ve doğrulma verilerindeki örüntüleri öğrenmeye başladığını ve tahminlerini iyileştirdiğini göstermektedir.

Eğitimin doğruluk metrikleri için Dice ve IoU metrikleri kullanılmıştır. Dice katsayısı, iki görüntü arasındaki kesişim ve birleşimi ölçen bir metrik olduğunu, IoU ise iki görüntü arasındaki örtüşme durumunu ölçen metrik olduğu ifade edilmişti. Her iki metrik için de yüksek katsayılar olması, modelin doğru segmentasyonlar ürettiğini ifade etmektedir. Dice metriği sonucunda 10 epokta 0,5480; 25 epokta 0,8066; 50 epokta, 0,8768 ve 100 epokta 0,9207 olarak elde edilmiştir. IoU metriği sonucunda ise 10 epokta 0,3514; 25 epokta 0,6436; 50 epokta, 0,7504 ve 100 epokta 0,8315 sonuçları bulunmuştur. Bu doğruluk metriklerine bakıldığında segmentasyon işleminin oldukça iyi olduğu görülmektedir.

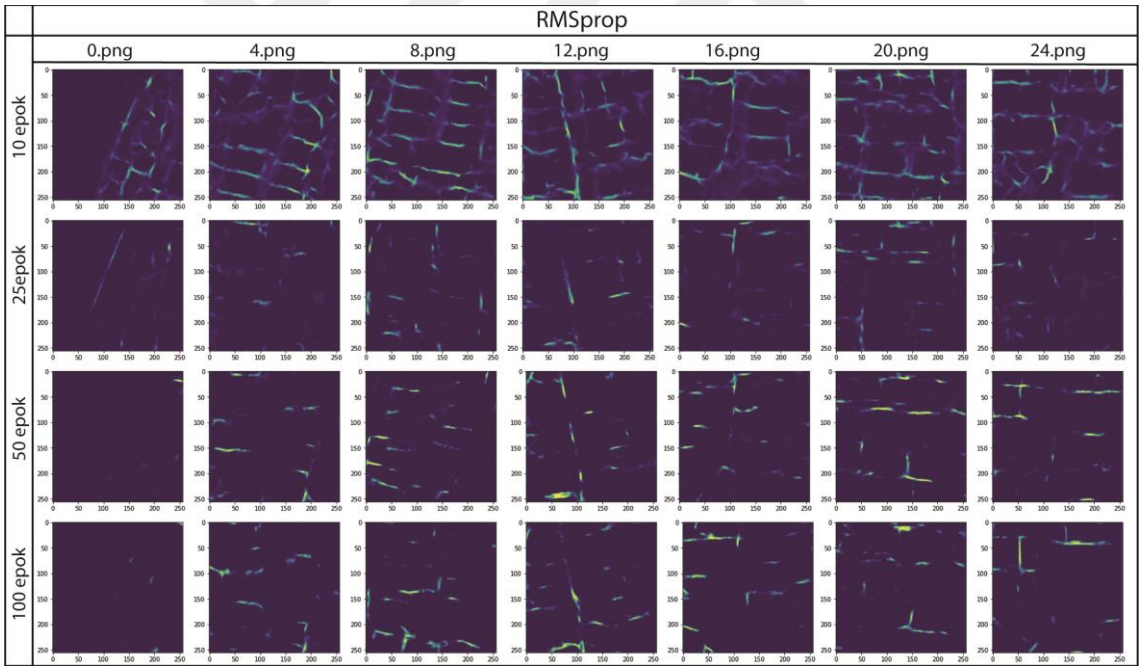
Kayıplar azalmakta, Dice katsayıları ve loU katsayıları artmaktadır. Bu, modelin yol adam optimizasyonu görevi için uygun olduğunu gösterir. Şekil4.31’de 7 adet görüntünün farklı epok sonuncunda tahmin edilen yol ağlarını ait görüntüleri verilmiştir.



Şekil 4.30. Yol ağı Adam için 10,25,50 ve 100 epoktaki Kayıp, Dice ve IoU grafikleri



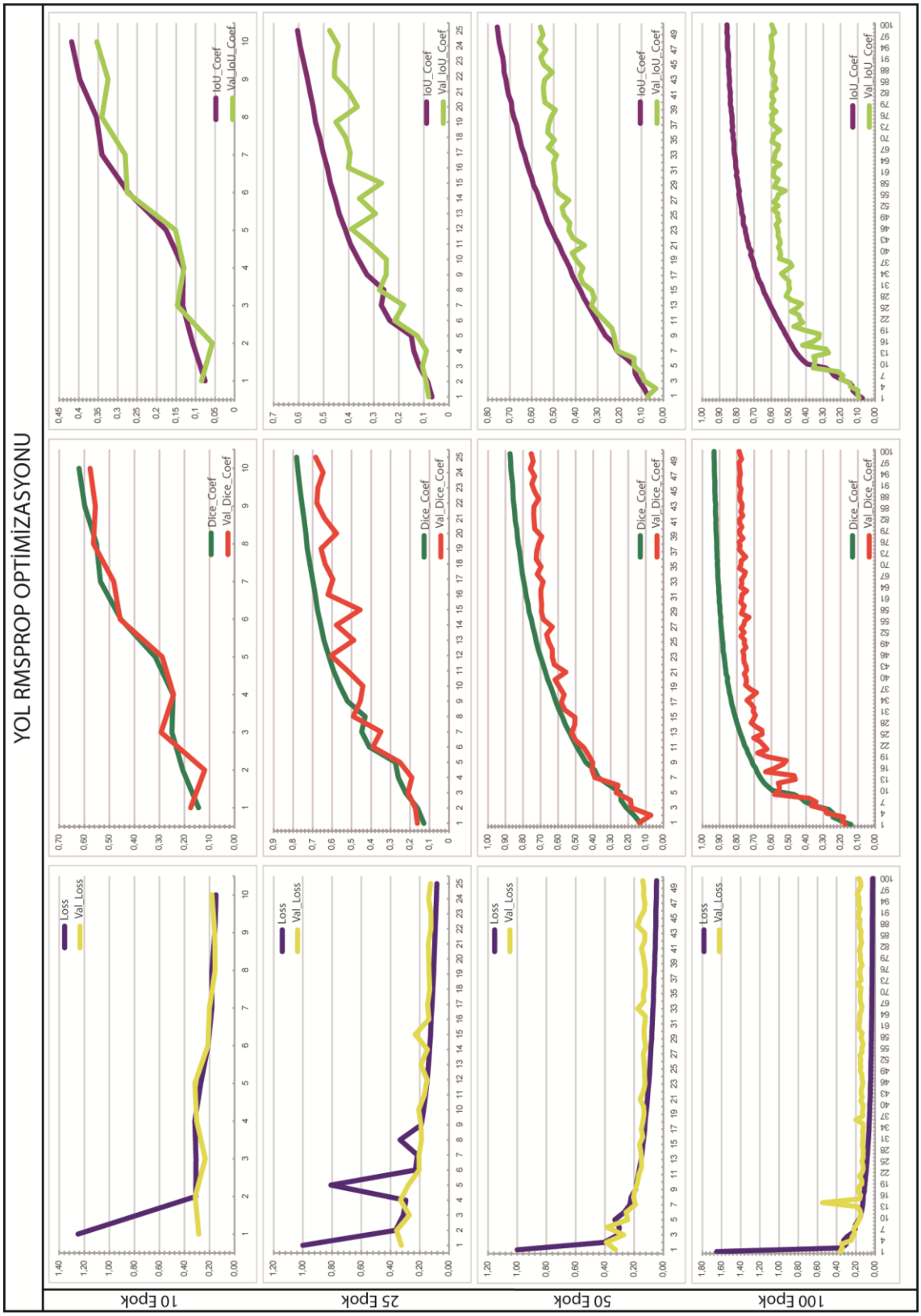
Şekil 4.31. Yol ağı Adam optimizasyonu sonunda elde edilen tahmin görüntüleri



Şekil 4.32. Yol ağı RMSprop optimizasyonu sonunda elde edilen tahmin görüntüleri

Şekil 4.33'te RMSprop optimizasyonu için kayıp ve doğruluk metriklerinin grafiklerine yer verilmiştir. Kayıp fonksiyonu 10 epok sonunda 0,1490; 25 epokta 0,0834; 50epokta 0,0460 ve 100 epokta 0,0260 seviyelerine kadar düşmüştür. Kayıp başlangıçta 0,1490 iken, 100 epok sonunda 0,03260 seviyelerine kadar düşmüştür. İstenilen bir durum söz konusudur. Dice metriği sonucunda 10 epokta 0,6212; 25 epokta 0,7819; 50

epokta, 0,8744 ve 100 epokta 0,9120 olarak elde edilmiştir. IoU metriği sonucunda ise 10 epokta 0,4181; 25 epokta 0,6046; 50 epokta, 0,7570 ve 100 epokta 0,8272 sonuçları bulunmuştur.



Şekil 4.33. Yol ağı RMSprop için 10,25,50 ve 100 epoktaki Kayıp, Dice ve IoU grafikleri

Bu doğruluk metriklerine bakıldığında Adam optimizasyonunda olduğu segmentasyon işleminin oldukça iyi olduğu görülmektedir. Fakat Adam optimizasyonunun metrik değerleri RMSprop'a kıyasla az daha iyi sonuçlar vermiştir. Binaların segmentasyonunda olduğu gibi yol ağlarının çıkarılmasında Adam optimizasyonu kullanılması daha iyi olduğu söylenebilir.

4.7. Tartışma

Çalışmanın sonucunda, U-net modelinin performansının öğrenme oranı, batch boyutu, epoch sayısı, eğitim verisinin miktarı ve optimizasyon parametrelerindeki değişikliklere göre farklılık gösterdiği belirlenmiştir. U-net modeli hem yol ağlarında hem de binaların çıkarımında kullanılmıştır. Binalarda eğitim verisi 1000 ve 100 epok sonunda Adam optimizasyonu kullanıldığında %82.23 doğruluk, %81.89 IoU ve %91.57 dice benzerliği sonucu elde edilmiştir. Bu metrikleri en yakın optimizasyon ise RMSprop'tur. Bu optimizasyon da elde edilen sonuçlar, %78.76 doğruluk, %70.85 IoU ve %85.05 dice benzerliğidir. Diğer optimizasyonlar verinin eğitimi için daha fazla epok sayısına ihtiyaç duyulmaktadır. Hem süre bakımından hem de doğruluk bakımından incelendiğin de Adam ve RMSprop optimizasyonlarının kısa sürede yüksek metrik değerlerine ulaştığı görülmektedir. Çalışmada elde edilen bulgular literatürdeki diğer çalışmalarla kıyaslandığında oldukça iyi sonuçlar verildiği görülmektedir. Yapılan çalışmalar ve metrik değerleri sunulan Çizelge 4.12'de model mimarisi, Çizelge 13'de kullanılan optimizasyon ve Çizelge 14'de epok durumunun önemi ve bu tez kapsamındaki sonuçlar kıyaslanmıştır. Çizelge 12'in ilk satırında sunulan metrikler, U-Net modelinin bina tespiti için Adam optimizasyon algoritması, 1e-3 öğrenme oranı, 16 batch boyutu ve 100 epok kullanılarak elde edilen sonuçları göstermektedir.

Çizelge 4.12. Literatürde bina çıkarımında kullanılan modeller

Çalışma	Model	Genel Doğruluk	IoU	Dice
Bu tez çalışması	U-net	%82.23	%81.89	%91.57
Atik ve ark. (2022)	ResNet-50	--	%67	--
Bischke ve ark. (2019)	SegNet	%74	--	--
Gibril ve ark. (2024)	YOLACT	%81	--	--
Q. Li ve ark. (2021)	FCN-8s	--	%79.33	--
Q. Li ve ark. (2021)	FC-DenseNet	--	%80.81	--

Çizelge 4.12'de bina çıkarımında en yüksek doğruluk bu çalışmada kullanılan U-net mimarisinde olduğu tespit edilmiştir. U-Net mimarisi, literatürdeki çeşitli çalışmalarla karşılaştırıldığında, uzaktan algılama görüntülerinden bina çıkarma görevinde yüksek bir dayanıklılık sergileyerek üstün performansını kanıtlamıştır. Farklı optimizasyon sonuçlarının etkisi Çizelge 13'te ele alınmıştır ve ilk iki satırında sunulan metrikler, U-Net modelinin bina tespiti için Adam ve RMSprop optimizasyon algoritması, 1e-3 öğrenme oranı, 16 batch boyutu ve 100 epok kullanılarak elde edilen sonuçları göstermektedir.

Çizelge 4.13. Literatürde bina çıkarımında kullanılan optimizasyon çeşitleri

Çalışma	**Optimizasyon**	**Genel Doğruluk**	**IoU**	**Dice**
Bu tez çalışması	Adam	%82.23	%81.89	%91.57
Bu tez çalışması	RMSprop	%78.76	%70.85	%85.08
Atik ve ark. (2022)	SGD	--	%67.60	--
Dunaeva (2019)	RMSprop		%81	
Mahmoud ve ark. (2021)	RMSprop	%78.9	--	--
Huang ve ark. (2021)	Adam	--	%88.32	--
Xu ve ark. (2021)	RAdam	--	%74.96	--

Huang ve ark. (2021) çalışmasında, Adam optimizasyon yöntemi kullanılmış ve IoU %88.32 olarak rapor etmişlerdir. Bu, Adam'ın çeşitli uygulamalarda yüksek IoU değerleri elde edebildiğini göstermektedir. Atik ve ark. (2022) çalışmasında ise, SGD optimizasyonunu tercih etmişlerdir ve %67.60 IoU skoruny elde etmişlerdir. Bu sonuç, SGD'nin diğer optimizasyon yöntemlerine göre daha düşük performans gösterebileceğini işaret etmektedir. Bir diğer optimizasyon ise, RAdam'dır. Xu ve ark. (2021) bu optimizasyonu yaptıkları çalışmada kullanmışlardır. %74.96 IoU skoru elde ederek, optimizasyonun rekabetçi bir performans sergileyebileceğini göstermişlerdir.

Farklı epok seçimlerinin sonuçlara etkisi Çizelge 14'de belirtilmiştir. Çizelge 14'ün ilk dört satırında sunulan metrikler, U-Net modelinin bina tespiti için Adam optimizasyon algoritması, 1e-3 öğrenme oranı, 16 batch boyutu ile 10, 25, 50 ve 100 epok kullanılarak elde edilen sonuçları göstermektedir.

Çizelge 4.14. Literatürde bina çıkarımında farklı epok değerlerinin kullanımı ve başarı metrikleri

Çalışma	Epok	Genel Doğruluk	IoU	Dice
Bu tez çalışması	10	%48.63	%11.84	%28.09
Bu tez çalışması	25	%56.44	%20.88	%48.36
Bu tez çalışması	50	%64.52	%42.90	%72.21
Bu tez çalışması	100	%82.23	%81.89	%91.57
Huang ve ark. (2021)	80		%88.32	
Wagner ve ark. (2020)	500	%97.67	%58.2	--
Srivastava ve ark. (2024)	100	%90.81	%82.97	%90.39
Yuxuan Li ve ark. (2024)	100	%94.77	%75.53	--

Genel olarak çizelge yorumlandığında, epok sayısının artışı modelin performansında belirgin bir iyileşme olduğunu görülmektedir. Ancak, her çalışma ve veri seti için optimal epok sayısı farklı olabilmesi söz konusudur. Çünkü, epok seçimi, modele, eğitim sayısına ya da probleme bağlı olarak değişebilir. Epok sayısının belirlenmesinde dikkatli bir dengeleme yapılmalı ve aşırı öğrenme gibi durumları göz önünde bulundurulmalıdır.

Algoritmanın işlem hızı, özellikle geniş ölçekli uzaktan algılama görüntülerinin analizinde kritik bir öneme sahip olduğu için, modeller eğitim ve test süreleri açısından da değerlendirilmiştir. Eğitim süreleri incelendiğinde tüm veri kümelerindeki en hızlı optimizasyon SGD'dir. En yavaş optimizasyon ise Adadelta da gerçekleşmiştir. Veri kümesi büyüdükçe eğitim için geçen süre de orantılı olarak arttığı tespit edilmiştir.

Sonuç olarak, U-Net mimarisinin kullanımı sırasında optimizasyon yöntemlerinin seçimi, öğrenme oranları ve batch boyutu parametrelerinin belirlenmesinin kritik olduğu sonucuna varılmıştır. Ayrıca, U-Net mimarisinin bina tespitinde etkili sonuçlar verdiği gözlemlenmiştir. Ek olarak, epoch sayısının ve eğitim verisinin artışıyla birlikte modelin daha belirgin hale geldiği ortaya çıkmıştır.

Çalışmanın devamında yol ağları çıkarım işlemi gerçekleştirilmiştir. Yol ağlarında eğitim verisi 1000 ve 100 epok sonunda Adam optimizasyonu kullanıldığında %86.14 doğruluk, %72.88 IoU ve %84.79 dice benzerliği sonucu elde edilmiştir. Çizelge 4.15'te model yapısı, Çizelge 16'da kullanılan optimizasyon yöntemi ve Çizelge 17'de ise epok sayısının önemi ile bu çalışmada elde edilen sonuçlar karşılaştırılmıştır. Çizelge15'in ilk satırında yer alan metrikler, U-Net modelinin yol ağları için Adam optimizasyon algoritması, 1e-3 öğrenme oranı, 16 batch boyutu ve 100 epok kullanılarak elde edilen sonuçları göstermektedir.

Çizelge 4.15. Literatürde yol çıkarımında kullanılan farklı algoritmalar ve başarı metrikleri

Çalışma	**Model**	**Genel Doğruluk**	**IoU**	**Dice**
Bu tez çalışması	U-net	%82.23	%81.89	%91.57
M. Wu ve ark. (2019)	LinkNet, D-LinkNet AD-LinkNet	--	LinkNet=%63 DLinkNet=%64.12 ADLinkNet=%78.5	--
B. Liu ve ark. (2023)	LDANet,	--	%76.21	--
X. Liu ve ark. (2023)	RoadFormer	--	%70.86	--
Lourenço ve ark. (2023)	DeepLabV3+	%86.83	--	--
Patil ve ark. (2022)	U - Net	--	--	%87
Hao ve ark. (2023)	U-Net, DeepLabv3+, PSPNet	--	U-Net=%49.26 DeepLabv3+=%70.33 PSPNet=%72.86	--
Ozturk ve ark. (2020)	U-Net, FCN	U-Net = %96.18 FCN=%97.69	--	--
Dewangan and Sahu (2021)	RCNet	%99,90	--	--

Çizelge 4.15'te, farklı derin öğrenme modellerinin çeşitli performans metrikleri açısından karşılaştırılmasını içermektedir. Genel olarak, yol ağlarının belirlenmesinde yaygın olarak kullanılan U-Net, DeepLabV3+, PSPNet ve diğer modellerin doğruluk, Intersection over Union (IoU) ve Dice katsayısı gibi metrikler üzerinden performansları değerlendirilmiştir. Literatürdeki çalışmalara göre %50 ve üzerinde bir başarı oranı olduğu görülmektedir. Dewangan and Sahu (2021) kullanmış oldukları RCNet modeli, %99.90 doğruluk oranı ile en yüksek performansı göstermektedir. Genel olarak, U-Net modeli, yüksek doğruluk ve Dice katsayısı ile öne çıkmaktadır. Fakat, Sariturk and Seker (2023) yaptıkları çalışmada elde ettikleri bulgular FCN modelinin başarı oranının %1.51 U-Net'e göre daha yüksek olduğunu ifade etmektedir. Doğruluk ve dice metriklerinin yanı sıra, IoU değerleri açısından AD-LinkNet ve LDANet modelleri dikkat çekmektedir.

Çizelge 4.16. Literatürde yol çıkarımında kullanılan farklı optimizasyonlar ve başarı metrikleri

Çalışma	Optimizasyon	Genel Doğruluk	IoU	Dice
Bu tez çalışması	Adam	%82.23	%81.89	%91.57
Junaid ve ark. (2020)	SGD ve RMSprop	--	%97.1	--
M. Wu ve ark. (2019)	Adam ve RMSProp	--	%78.5	--
Dewangan and Sahu (2021)	SGD, Adadelta, RMSprop, Adagrad, Adam, Nadam, Adamax	%99,90	--	--
Lourenço ve ark. (2023)	Adam	%86.83	--	--
Patil ve ark. (2022)	Adam	--	--	%87

Çizelgede ifade edildiği üzere, çeşitli çalışmalarda farklı optimizasyon yöntemleri kullanmıştır. Dewangan and Sahu (2021) çalışmasında %99.90 gibi oldukça yüksek bir doğruluk oranı bildirilmiştir. Lourenço ve ark. (2023) %86.83 doğruluk oranıyla bu tez çalışmasından (%82.23) daha iyi bir performans göstermiştir. Ayrıca, Dewangan and Sahu (2021) çalışmasında genel doğruluk açısından en iyi performansı sergilerken, Junaid ve ark. (2020) çalışması IoU açısından üstünlük sağlamıştır. Çalışmaların genelinde Adam optimizasyon yöntemi yaygın olarak kullanılmış ve genellikle iyi performans göstermiştir.

Bina ve yol ağlarının çıkarımı kıyaslandığında, U-Net mimarisinin bina tespitinde daha iyi sonuçlar elde edildiği görülmüştür. Yol ağlarında iyi sonuçlar vermediği tespit edilmiştir.

5. SONUÇLAR VE ÖNERİLER

Görüntülerin segmentasyonunda derin öğrenme mimarileri geçmişe kıyasla günümüzde sıkça kullanılan yöntemlerdendir. Bu tez çalışmasında, yeryüzünün zaman bağlı olarak değişimi sonucunda kentsel planlama, coğrafi bilgi sistemleri ve yol ağları yönetimi gibi alanlarda, doğru ve güncel bir harita, çeşitli planlama ve analiz faaliyetlerinde kullanılabilmesi amacıyla binalar ve yol ağlarının hızlı bir şekilde belirlenmesi amacıyla araştırmalar gerçekleştirilmiştir.

Belirlenen problemin çözümü için, Göktürk-1 uydu görüntüsü kullanarak bina ve yol ağlarını tespit etme uygulamasında derin öğrenme algoritmasının U-Net modeli incelenmiştir. Inria hazır veri setinden yararlanarak model eğitimi gerçekleştirilmiş olup, Göktürk-1 uydu görüntülerine uygulanmıştır. Test verilerinin hazır veri seti yerine Göktürk-1 uydusu görüntüleri kullanılarak, Ankara'da bulunan binaların geometrik yapısı, yıllar içerisindeki durumları ve yoğunlukları hakkında çıkarım yapılması mümkün olduğu görülmektedir.

Derin öğrenme teknikleri, genellikle geniş miktarda doğru etiket verisine ihtiyaç duyar; bu nedenle, uzaktan algılama görüntülerinin etiketlenmesi, genellikle manuel olarak titizlikle gerçekleştirilmesi gereken bir süreçtir. Binalar, yollar ve şehirler gibi belirli alanlarda uzmanlaşmış kişiler tarafından maskelenmiş ve kamuya açık uzaktan algılama veri kümeleri sayesinde, farklı uydu görüntüleri üzerinde hızlı ve otomatik çıkarım işlemleri gerçekleştirilebilmektedir. Bununla birlikte, hidroloji, buzullar, mahsuller ve benzeri doğal araştırma nesneleri için kamuya açık uzaktan algılama veri kümelerinin kısıtlı olması, bu alanlarda otomatik analiz ve çıkarımın geliştirilmesini sınırlamaktadır. Maskelenmiş görüntülerin sağlanması ve algoritmanın doğruluğunun artırılmasıyla birlikte, uzaktan algılama görüntülerinin segmentasyonunda başarılı sonuçlar elde etmek için algoritmanın daha fazla anlamsal bilgi edinmesine yardımcı olacak şekilde birden fazla veri kaynağının entegrasyonu da önem arz etmektedir.

Yol ağlarının tamamı belirlenmesi mümkün olmamıştır. Bu durumun temel sebebi, görüntülerinin alımı sırasında yol kenarında ya da refüj kısmında ağaçların bulunması, yol üzerinde araçların seyir halinde olması ya da yol kenarlarına park edilmiş olmasının negatif etkileri bulunmaktadır. Benzer şekilde görüntü alımında mevsimlerin de sonuçlara etkisi bulunduğu tespit edilmiştir. Özellikle yaz ve kış mevsiminin etkisi büyüktür. Çünkü, 2020 ve 2021 yıllarındaki görüntüler kış mevsimi döneminde alınmış ve sonuç görüntüleri anlamlı bilgiler üretmemiştir. Oysaki 2018 ve 2019 yılındaki yaz

mevsimine denk gelen dönemler de alınan görüntülerde binaların çıkarımı daha anlamlı olmuştur.

Benzer şekilde binaların etrafındaki yapay özelliklerin ve gölgelerin etkisi, örneğin bazı binaların gri değeri, gölgelerinin gri değerine yakın olduğundan aralarındaki sınırı ayırt etmek zordur. Uzaktan algılama görüntülerinin farklı çekim açıları ve süreleri nedeniyle her yapı taşı farklı türde gölgeler üretir ve bu da binaların tanınmasını etkiler. Bazı tanıma ve çıkarma algoritmaları, geometrik sınıra dayalı yöntemin doku özelliklerinden, tek bir bina arasındaki mekânsal ilişki özelliklerinden tam olarak faydalanmaması gibi bazı kusurlara sahiptir. Ayrıca bina yapısının karmaşıklığı nedeniyle bina, genellikle diğer insan yapımı hedefler veya doğal hedefler tarafından gizlenir ve bu da binanın tanımlanmasında bir müdahale faktörü haline gelir.

Yapılan çalışma sonucunda derin öğrenmenin özelliklerine göre eğitim setlerinde bulunan verinin miktarı ve verilerin zengin özelliklere (geometrik ve renk) sahip olması doğruluk oranını daha da arttırdığı görülmektedir. Bu çalışma, U-net mimarisi kullanarak görüntü segmentasyonunda batch boyutlarının optimize edilmesi için önemli bir referans oluştururken, batch boyutlarının seçiminde performans ve hesaplama kaynakları dengesinin göz önünde bulundurulması gerektiğini vurgulamaktadır.

Derin öğrenme algoritmaları, özellikle karmaşık algoritmaların eğitimi sırasında, bilgisayar GPU'larının yüksek işlem gücü ve bellek kapasitesinden faydalanarak hesaplama ve depolama gereksinimlerini karşılamak için yüksek performanslı donanım kaynaklarına ihtiyaç duymaktadır. Bu nedenle kullanılan donanım görüntü işlemek için uygun özelliklere sahip olmalıdır. Çünkü donanım, model eğitimi için harcanan süreyi etkilemektedir. Optimizasyon algoritmasını sadece süre yönünden seçmek yanlış olacaktır. Her bir optimizasyonun birbirlerine göre avantajları ve dezavantajları vardır. Bu nedenle uygun olanı seçmek, genellikle veri seti ve modelini özelliklerine bağlıdır. Algoritmanın hızı, özellikle büyük ölçekli uzaktan algılama görüntülerinin işlenmesinde büyük önem taşımaktadır. Bu nedenle çalışmada model eğitimi için geçen süre ve doğruluk kıyaslaması yapılmıştır. Epok sayısının artması, batch boyutunun artması, süre ile doğru orantılıdır. Bu parametreler arttıkça modelin eğitimi için harcanan süre de artmaktadır. Öğrenme oranı için net bir şey söylemek mümkün değildir. Çünkü 1e-4 ve 1e-3 öğrenme oranlarında genelde 2-3 dk farklılıklar bulunmaktadır.

Derin öğrenmenin uzaktan algılama görüntülerinden bina ve yol çıkarımı konusunda umut verici sonuçlar sunduğu kabul edilse de bu alanda hala geliştirilmesi

gereken birçok nokta bulunmaktadır. Özellikle yol ağlarının tam hatlarıyla belirlenememiş olması büyük bir eksikliktir.

Geleneksel yöntemlerle karşılaştırıldığında, derin öğrenme yöntemleri yol çıkarmanın otomasyonunu, uyumluluğunu ve doğruluğunu büyük ölçüde geliştirdi. Sonuç olarak, güçlü uzaysal heterojenliğe sahip görüntüler işlenirken, çıkarılan özelliklerde hala birçok hatalı çıkarım ve yol kısımlarında eksiklikler bulunmaktadır.

Bu çalışmayla bir bölgedeki bina ve yol yoğunluğunu belirlendikten sonra, yeşil alanları korumak için planlama kararlarını desteklemek için kullanılabilmesini sağlayacağı da düşünülmektedir. Ya da deprem gibi afetlerin yaşanması durumunda bina tespitinde kullanılabileceği düşünülmektedir. Yol ağlarındaki problemlerin çözülmesi durumunda trafik akışını optimize etmek için ya da yol ağlarının belirlenmesi gibi çalışmalarda altlık olarak kullanılabilmesi muhtemeldir.

Gelecekte yapılacak çalışmalarda, Göktürk-1 uydu görüntülerinden elde edilecek coğrafi veri setinin oluşturulması ve bu veri setinin eğitim verileri arasına dahil edilmesi planlanmaktadır. Bu yaklaşım, çeşitli bina ve yol verilerinin kapsamını genişleterek, modelin genel performansını ve doğruluğunu artırmayı hedeflemektedir. Özellikle yüksek çözünürlüklü uydu görüntülerinin kullanılması, detaylı ve doğru nesne tespiti sağlayarak, modelin genelleme kabiliyetini güçlendirecektir. Böylece, elde edilen verilerle zenginleştirilen eğitim seti sayesinde, daha etkili ve güvenilir sonuçlar elde edilebilecektir.

KAYNAKLAR

Akçura, T., 1971, Ankara: Türkiye Cumhuriyeti'nin başkenti hakkında monografik bir araştırma, *(No Title)*.

Ali, M. ve Clausi, D., 2001, Using the Canny edge detector for feature extraction and enhancement of remote sensing images, *Paper presented at the IGARSS 2001. Scanning the Present and Resolving the Future. Proceedings. IEEE 2001 International Geoscience and Remote Sensing Symposium (Cat. No. 01CH37217)*.

Arasan, G., Yılmaz, A., Fırat, O., Avşar, E., Güner, H., Ayğan, K. ve Yüce, D., 2020, Accuracy assessments of Göktürk-1 satellite imagery, *International Journal of Engineering and Geosciences*, 5(3), 160-168. DOI:10.26833/ijeg.650899

Arıkan, D. ve Yıldız, F., 2023, Göktürk-1 Uydu Görüntülerinden U-Net Modeli Kullanılarak Binaların Segmentasyonu, *Türkiye Uzaktan Algılama Dergisi*, 5(1), 50-58. DOI:10.51489/tuzal.1300939

Atik, S. O., Atik, M. E. ve Ipbuker, C., 2022, Comparative research on different backbone architectures of DeepLabV3+ for building segmentation, *Journal of applied remote sensing*, 16(2), 024510-024510. DOI:10.1117/1.JRS.16.024510

Aytekin, G. ve Topan, H., 2022, Üçlü Bindirmeli Göktürk-1 Uydu Görüntülerinin Konum Doğruluğunun Zonguldak Test Alanında Algılayıcıya Bağımlı Yöneltme Modeli ile Belirlenmesi, *Harita Dergisi*, 168, 13-27.

Badrinarayanan, V., Kendall, A. ve Cipolla, R., 2017, Segnet: A deep convolutional encoder-decoder architecture for image segmentation, *IEEE transactions on pattern analysis and machine intelligence*, 39(12), 2481-2495. DOI:10.1109/TPAMI.2016.2644615

Baltsavias, E. P., 2004, Object extraction and revision by image analysis using existing geodata and knowledge: current status and steps towards operational systems, *ISPRS Journal of photogrammetry and remote sensing*, 58(3-4), 129-151. DOI:10.1016/j.isprsjprs.2003.09.002

Batuman, B., 2013, City profile: Ankara, *Cities*, 31, 578-590. DOI:10.1016/j.cities.2012.05.016

Bayar, R., 2020, Ankara Şehri Kentsel Büyüme Alanlarının Arazi Uygunluk Analizi, *Ankara Üniversitesi Dil ve Tarih-Coğrafya Fakültesi Dergisi*, 60(1), 39-59.

Bayramoğlu, Z., 2021, Nesneye Yönelik Kural Tabanlı Sınıflandırma ve Derin Öğrenme Yöntemleri ile Otomatik Yol Çıkarımı Olanaklarının Araştırılması, (Yüksek Lisans), *Yıldız Teknik Üniversitesi*.

Belgiu, M. ve Drăguţ, L., 2016, Random forest in remote sensing: A review of applications and future directions, *ISPRS Journal of photogrammetry and remote sensing*, 114, 24-31. DOI:10.1016/j.isprsjprs.2016.01.011

Bischke, B., Helber, P., Folz, J., Borth, D. ve Dengel, A., 2019, Multi-task learning for segmentation of building footprints with deep neural networks, *Paper presented at the 2019 IEEE International Conference on Image Processing (ICIP)*.

Bozkurt, S., 2018, Derin öğrenme algoritmaları kullanılarak çay alanlarının otomatik segmentasyonu, (Yüksek Lisans Tezi.), *Yıldız Teknik Üniversitesi*.

Brenner, C., 2005, Building reconstruction from images and laser scanning, *International Journal of Applied Earth Observation and Geoinformation*, 6(3-4), 187-198. DOI:10.1016/j.jag.2004.10.006

Buchanan, B. G., 2005, A (very) brief history of artificial intelligence, *Ai Magazine*, 26(4), 53-53. DOI:10.1609/aimag.v26i4.1848

Chaudhuri, D., Kushwaha, N. K. ve Samal, A., 2012, Semi-automated road detection from high resolution satellite images by directional morphological enhancement and segmentation techniques, *IEEE Journal of selected topics in applied earth observations and remote sensing*, 5(5), 1538-1544. DOI:10.1109/JSTARS.2012.2199085

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K. ve Yuille, A. L., 2017, Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, *IEEE transactions on pattern analysis and machine intelligence*, 40(4), 834-848. DOI:10.1109/TPAMI.2017.2699184

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F. ve Adam, H., 2018, Encoder-decoder with atrous separable convolution for semantic image segmentation, *Paper presented at the Proceedings of the European conference on computer vision (ECCV)*.

Courtrai, L. ve Lefèvre, S., 2016, Morphological path filtering at the region scale for efficient and robust road network extraction from satellite imagery, *Pattern recognition letters*, 83, 195-204. DOI:10.1016/j.patrec.2016.05.014

Çam, A., 2023, Bina Enerji Modellemesi yöntemiyle konut yapısı enerji analizi: Ankara Yeşiltepe Yapı Kooperatifi örneği, (Yüksek Lisans Tezi), *Necmettin Erbakan Üniversitesi*

Dai, J., Du, Y., Zhu, T., Wang, Y. ve Gao, L., 2019, Multiscale residual convolution neural network and sector descriptor-based road detection method, *IEEE Access*, 7, 173377-173392. DOI:10.1109/ACCESS.2019.2956725

de Carvalho, S. D., de Melo, F. R., Flôres, E. L., Pires, S. R. ve Loja, L. F. B., 2020, Intelligent tutoring system using expert knowledge and Kohonen maps with automated training, *Neural Computing and Applications*, 32(17), 13577-13589. DOI:10.1007/s00521-020-04767-0

Dewangan, D. ve Sahu, S., 2021, RCNet: road classification convolutional neural networks for intelligent vehicle system. Intell Serv Robot 14 (2): 199–214. DOI:10.1007/s11370-020-00343-6

Duchi, J., Hazan, E. ve Singer, Y., 2011, Adaptive subgradient methods for online learning and stochastic optimization, *Journal of machine learning research*, 12(7).

Dunaeva, A., 2019, Building Footprint Extraction from Stereo Satellite Imagery Using Convolutional Neural Networks, *Paper presented at the 2019 International Multi-Conference on Engineering, Computer and Information Sciences (SIBIRCON)*.

Eerapu, K. K., Ashwath, B., Lal, S., Dell'Acqua, F. ve Dhan, A. N., 2019, Dense refinement residual network for road extraction from aerial imagery data, *IEEE Access*, 7, 151764-151782. DOI:10.1109/ACCESS.2019.2928882

El-Mekawy, M., Östman, A. ve Hijazi, I., 2012, A unified building model for 3D urban GIS, *ISPRS International Journal of Geo-Information*, 1(2), 120-145. DOI:10.3390/ijgi1020120

Emek, R. A., 2022, Derin öğrenme metotları kullanılarak SAR (Sentetik Açıklıklı Radar) görüntülerinden bina tespiti, (Yüksek Lisans Tezi), *Akdeniz Üniversitesi*.

Erhandı, B., 2020, Derin öğrenme ile metin özetleme, (Yüksek Lisans Tezi), *Sakarya Üniversitesi*.

Fu, Y., Liu, K., Shen, Z., Deng, J., Gan, M., Liu, X., Lu, D. ve Wang, K., 2019, Mapping impervious surfaces in town–rural transition belts using China's GF-2 imagery and object-based deep CNNs, *Remote Sensing*, 11(3), 280. DOI:10.3390/rs11030280

Gao, F., Huang, T., Sun, J., Wang, J., Hussain, A. ve Yang, E., 2019, A new algorithm for SAR image target recognition based on an improved deep convolutional neural network, *Cognitive Computation*, 11, 809-824. DOI:10.1007/s12559-018-9563-z

Gao, J., Wang, H. ve Shen, H., 2020, Task failure prediction in cloud data centers using deep learning, *IEEE transactions on services computing*, 15(3), 1411-1422. DOI:10.1109/TSC.2020.2993728

Gazel, S. ve Bati, C. T., 2019, Derin sinir ağları ile en iyi modelin belirlenmesi: mantar verileri üzerine keras uygulaması, *Yuzuncu Yıl University Journal of Agricultural Sciences*, 29(3), 406-417.

Ghosh, A., Ehrlich, M., Shah, S., Davis, L. S. ve Chellappa, R., 2018, Stacked U-Nets for ground material segmentation in remote sensing imagery, *Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition workshops*.

Gibril, M. B. A., Al-Ruzouq, R., Bolcek, J., Shanableh, A. ve Jena, R., 2024, Building Extraction from Satellite Images Using Mask R-CNN and Swin Transformer, *Paper presented at the 2024 34th International Conference Radioelektronika (RADIOELEKTRONIKA)*.

Guerrero-Ibañez, J., Contreras-Castillo, J. ve Zeadally, S., 2021, Deep learning support for intelligent transportation systems, *Transactions on Emerging Telecommunications Technologies*, 32(3), e4169. DOI:10.1002/ett.4169

Gui, Y., Li, X., Li, W. ve Yue, A., 2018, Multi-branch regression network for building classification using remote sensing images, *Paper presented at the 2018 10th IAPR Workshop on Pattern Recognition in Remote Sensing (PRRS)*.

Guo, Y., Zhou, M., Wang, Y., Wu, G. ve Shibasaki, R., 2022, Learn to Be Clear and Colorful: An End-to-End Network for Panchromatic Image Enhancement, *IEEE Geoscience and Remote Sensing Letters*, 19, 1-5. DOI:10.1109/LGRS.2022.3142994

Gupta, A., Dixit, M., Sharma, V. ve Nand, P., 2022, Road Extraction from Medium Resolution Satellite Images using Deep Learning Techniques, *Paper presented at the 2022 4th International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)*.

h5py, 2024, https://docs.h5py.org/en/stable/ [Ziyaret Tarihi: 2024].

Hänsch, R. ve Hellwich, O., 2010, Random forests for building detection in polarimetric sar data, *Paper presented at the 2010 IEEE International Geoscience and Remote Sensing Symposium*.

Hao, X., Yin, L., Li, X., Zhang, L. ve Yang, R., 2023, A Multi-Objective Semantic Segmentation Algorithm Based on Improved U-Net Networks, *Remote Sensing*, 15(7), 1838. DOI:10.3390/rs15071838

Hay, J. C., Lynch, B. E. ve Smith, D. R., 1960, Mark I perceptron operators' manual, *Cornell Aeronautical Lab., Buffalo, NY, Rept. No. VG-1196-G-5*.

He, H., Xu, H., Zhang, Y., Gao, K., Li, H., Ma, L. ve Li, J., 2022, Mask R-CNN based automated identification and extraction of oil well sites, *International Journal of Applied Earth Observation and Geoinformation*, 112, 102875. DOI:10.1016/j.jag.2022.102875

He, H., Yang, D., Wang, S., Wang, S. ve Li, Y., 2019, Road extraction by using atrous spatial pyramid pooling integrated encoder-decoder network and structural similarity loss, *Remote Sensing*, 11(9), 1015.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P. ve Sainath, T. N., 2012, Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups, *IEEE Signal processing magazine*, 29(6), 82-97. DOI:10.1109/MSP.2012.2205597

Hopfield, J. J. ve Tank, D. W., 1985, "Neural" computation of decisions in optimization problems, *Biological cybernetics*, 52(3), 141-152.

Hu, L., Zheng, J. ve Gao, F., 2011, A building extraction method using shadow in high resolution multispectral images, *Paper presented at the 2011 IEEE International Geoscience and Remote Sensing Symposium*.

Huang, H., Chen, Y. ve Wang, R., 2021, A lightweight network for building extraction from remote sensing images, *IEEE Transactions on Geoscience and Remote Sensing*, 60, 1-12. DOI:10.1109/TGRS.2021.3131331

İsa, A., 2023, Performance Evaluation of Jaccard-Dice Coefficient on Building Segmentation from High Resolution Satellite Images, *Balkan Journal of Electrical and Computer Engineering*, 11(1), 100-106. DOI:10.17694/bajece.1212563

Jaynes, C. O., Stolle, F. ve Collins, R. T., 1994, Task driven perceptual organization for extraction of rooftop polygons, *Paper presented at the Proceedings of 1994 IEEE Workshop on Applications of Computer Vision*.

Ji, S., Wei, S. ve Lu, M., 2018, Fully convolutional networks for multisource building extraction from an open aerial and satellite imagery data set, *IEEE Transactions on Geoscience and Remote Sensing*, 57(1), 574-586. DOI:10.1109/TGRS.2018.2858817

Jiang, B., An, X., Xu, S. ve Chen, Z., 2023, Intelligent image semantic segmentation: a review through deep learning techniques for remote sensing image analysis, *Journal of the Indian society of remote sensing*, 51(9), 1865-1878. DOI:10.1007/s12524-022-01496-w

Junaid, M., Ghafoor, M., Hassan, A., Khalid, S., Tariq, S. A., Ahmed, G. ve Zia, T., 2020, Multi-feature view-based shallow convolutional neural network for road segmentation, *IEEE Access*, 8, 36612-36623. DOI:10.1109/ACCESS.2020.2968965

Kalkan, K. ve Maktav, D., 2010, Nsne Tabanlı ve Piksel Tabanlı Sınıflandırma Yöntemlerinin Karşılaştırılması (IKONOS Örneği).

Kanal, L. N., 2003, Perceptron Encyclopedia of Computer Science (pp. 1383-1385).

Kattenborn, T., Eichel, J. ve Fassnacht, F. E., 2019, Convolutional Neural Networks enable efficient, accurate and fine-grained segmentation of plant species and communities from high-resolution UAV imagery, *Scientific reports*, 9(1), 17656. DOI:10.1038/s41598-019-53797-9

Kemal, A. ve Kilicarslan, S., 2019, Performance analysis of optimization algorithms on stacked autoencoder, *Paper presented at the 2019 3rd international symposium on multidisciplinary studies and innovative technologies (ISMSIT)*.

Keras, 2024, Python Kütüphanesi, https://keras.io/ [Ziyaret Tarihi: 09 Nisan 2024].

Kingma, D. P. ve Ba, J., 2014, Adam: A method for stochastic optimization, *arXiv preprint arXiv:1412.6980*. DOI:10.48550/arXiv.1412.6980

Kırcalı, K., 2022, Element eksikliği olan bölgelerdeki ürünlerin derin öğrenme ve İHA ile sınıflandırılması, (Yüksek Lisans Tezi), *Konya Teknik Üniversitesi*.

Krishnamachari, S. ve Chellappa, R., 1996, Delineating buildings by grouping lines with MRFs, *IEEE Transactions on image processing*, 5(1), 164-168. DOI:10.1109/83.481683

Krizhevsky, A., Sutskever, I. ve Hinton, G. E., 2017, ImageNet classification with deep convolutional neural networks, *Communications of the ACM*, 60(6), 84-90. DOI:10.1145/306538

Kussul, N., Lavreniuk, M., Skakun, S. ve Shelestov, A., 2017, Deep learning classification of land cover and crop types using remote sensing data, *IEEE Geoscience and Remote Sensing Letters*, 14(5), 778-782. DOI:10.1109/LGRS.2017.2681128

LeCun, Y., Bengio, Y. ve Hinton, G., 2015, Deep learning, *nature*, 521(7553), 436-444.

Li, Q., Mou, L., Hua, Y., Shi, Y. ve Zhu, X. X., 2021, Building footprint generation through convolutional neural networks with attraction field representation, *IEEE Transactions on Geoscience and Remote Sensing*, 60, 1-17. DOI:10.1109/TGRS.2021.3109844

Li, X., Yao, X. ve Fang, Y., 2018, Building-a-nets: Robust building extraction from high-resolution remote sensing images with adversarial networks, *IEEE Journal of selected topics in applied earth observations and remote sensing*, 11(10), 3680-3687. DOI:10.1109/JSTARS.2018.2865187

Li, Y., Guo, L., Rao, J., Xu, L. ve Jin, S., 2018, Road segmentation based on hybrid convolutional network for high-resolution visible remote sensing image, *IEEE Geoscience and Remote Sensing Letters*, 16(4), 613-617. DOI:10.1109/LGRS.2018.2878771

Li, Y., Hong, D., Li, C., Yao, J. ve Chanussot, J., 2024, HD-Net: High-resolution decoupled network for building footprint extraction via deeply supervised body and boundary decomposition, *ISPRS Journal of photogrammetry and remote sensing*, 209, 51-65. DOI:10.1016/j.isprsjprs.2024.01.022

Li, Y. ve Wu, H., 2008, Adaptive building edge detection by combining LiDAR data and aerial images, *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 37(Part B1), 197-202. DOI:10.1109/TGRS.2018.2870871

Li, Z., Wang, H. ve Liu, Y., 2023, Semantic segmentation of remote sensing image based on bilateral branch network, *The Visual Computer*, 1-22. DOI:10.1007/s00371-023-03011-9

Lin, Huertas ve Nevatia, 1994, Detection of buildings using perceptual grouping and shadows, *Paper presented at the 1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Lin, X., Zhang, J., Liu, Z. ve Shen, J., 2009, Semi-automatic road tracking by template matching and distance transform, *Paper presented at the 2009 Joint Urban Remote Sensing Event*.

Lin, X., Zhang, R. ve Shen, J., 2012, A template-matching based approach for extraction of roads from very high-resolution remotely sensed imagery, *International Journal of Image and Data Fusion*, 3(2), 149-168. DOI:10.1080/19479832.2011.642413

Liow, Y.-T. ve Pavlidis, T., 1990, Use of shadows for extracting buildings in aerial images, *Computer Vision, Graphics, and Image Processing*, 49(2), 242-277. DOI:10.1016/0734-189X(90)90139-M

Liu, B., Ding, J., Zou, J., Wang, J. ve Huang, S., 2023, LDANet: a lightweight dynamic addition network for rural road extraction from remote sensing images, *Remote Sensing*, 15(7), 1829. DOI:10.3390/rs15071829

Liu, S., Ye, H., Jin, K. ve Cheng, H., 2021, CT-UNet: Context-transfer-UNet for building segmentation in remote sensing images, *Neural Processing Letters*, 53, 4257-4277. DOI:10.1007/s11063-021-10592-w

Liu, X., Wang, Z., Wan, J., Zhang, J., Xi, Y., Liu, R. ve Miao, Q., 2023, RoadFormer: Road extraction using a Swin transformer combined with a spatial and channel separable convolution, *Remote Sensing*, 15(4), 1049. DOI:10.3390/rs15041049

Liu, Y., Yao, J., Lu, X., Xia, M., Wang, X. ve Liu, Y., 2018, RoadNet: Learning to comprehensively analyze road networks in complex urban scenes from high-resolution remotely sensed images, *IEEE Transactions on Geoscience and Remote Sensing*, 57(4), 2043-2056.

Long, J., Shelhamer, E. ve Darrell, T., 2015, Fully convolutional networks for semantic segmentation, *Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition*.

Lourenço, M., Estima, D., Oliveira, H., Oliveira, L. ve Mora, A., 2023, Automatic rural road centerline detection and extraction from aerial images for a forest fire decision support system, *Remote Sensing*, 15(1), 271. DOI:10.3390/rs15010271

Lu, T., Ming, D., Lin, X., Hong, Z., Bai, X. ve Fang, J., 2018, Detecting building edges from high spatial resolution remote sensing imagery using richer convolution features network, *Remote Sensing*, 10(9), 1496. DOI:10.3390/rs10091496

Luo, H., Wang, L., Shao, Z. ve Li, D., 2015, Development of a multi-scale object-based shadow detection method for high spatial resolution image, *Remote sensing letters*, 6(1), 59-68. DOI:10.1080/2150704X.2014.1001079

Maas, A. L., Hannun, A. Y. ve Ng, A. Y., 2013, Rectifier nonlinearities improve neural network acoustic models, *Paper presented at the Proc. icml*.

Madhu, G., Kautish, S., Alnowibet, K. A., Zawbaa, H. M. ve Mohamed, A. W., 2023, NIPUNA: A Novel Optimizer Activation Function for Deep Neural Networks, *Axioms*, 12(3), 246. DOI:10.3390/axioms12030246

Maggiori, E., Tarabalka, Y., Charpiat, G. ve Alliez, P., 2017, Can semantic labeling methods generalize to any city? the inria aerial image labeling benchmark, *Paper presented at the 2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*.

Mahmoud, A. S., Mohamed, S. A., Moustafa, M. S., El-Khorib, R. A., Abdelsalam, H. M. ve El-Khodary, I. A., 2021, Training compact change detection network for remote sensing imagery, *IEEE Access*, 9, 90366-90378. DOI:10.1109/ACCESS.2021.3089766

Maltezos, E., Doulamis, N., Doulamis, A. ve Ioannidis, C., 2017, Deep convolutional neural networks for building extraction from orthoimages and dense image matching point clouds, *Journal of applied remote sensing*, 11(4), 042620-042620. DOI:10.1117/1.JRS.11.042620

Matplotlib, 2024, https://matplotlib.org/ [Ziyaret Tarihi: 09 Nisan 2024].

McCarthy, J., Minsky, M. L., Rochester, N. ve Shannon, C. E., 2006, A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955, *Ai Magazine*, 27(4), 12-12. DOI:10.1609/aimag.v27i4.1904

Mena, J. B., 2003, State of the art on automatic road extraction for GIS update: a novel classification, *Pattern recognition letters*, 24(16), 3037-3058. DOI:10.1016/S0167-8655(03)00164-8

Metlek, S. ve Çetiner, H., 2021, Derin Öğrenme Matlab Ortamında Derin Öğrenme Uygulamaları: İksad Yayınevi.

Mikaeili, M., 2016, Kenar kavramının peyzaj mimarlığı açısından Ankara kenti örneğinde araştırılması, (Doktora (Phd)), *Ankara Üniversitesi*.

Ming, D.-P., Luo, J.-C., Shen, Z.-F., Wang, M. ve Sehng, H., 2005, Research on information extraction and target recognition from high resolution remote sensing image, *Cehui Kexue/ Science of Surveying and Mapping*, 30(3), 18-20.

Mohanty, S. P., Hughes, D. P. ve Salathé, M., 2016, Using deep learning for image-based plant disease detection, *Frontiers in plant science*, 7, 1419. DOI:10.3389/fpls.2016.01419

Muehlenbein, H., 2014, Artificial intelligence and neural networks the legacy of Alan Turing and John von Neumann, *Int J Comput*, 5(3), 10-20.

Newman, M. E., McLaren, K. P. ve Wilson, B. S., 2011, Comparing the effects of classification techniques on landscape-level assessments: pixel-based versus object-based classification, *International journal of remote sensing*, 32(14), 4055-4073. DOI:10.1080/01431161.2010.484432

Numpy, 2024, Python Kütüphanesi, https://numpy.org/ [Ziyaret Tarihi: 11 Nisan 2024].

Nwankpa, C., Ijomah, W., Gachagan, A. ve Marshall, S., 2018, Activation functions: Comparison of trends in practice and research for deep learning, *arXiv preprint arXiv:1811.03378*. DOI:10.48550/arXiv.1811.03378

Ok, A. O., 2013, Automated detection of buildings from single VHR multispectral images using shadow information and graph cuts, *ISPRS Journal of photogrammetry and remote sensing*, 86, 21-40. DOI:10.1016/j.isprsjprs.2013.09.004

Okan, V., 2023, Ankara ilinin turizm potansiyeli (Yüksek Lisans).

OpenCV, 2024, https://opencv.org/ [Ziyaret Tarihi: 2024].

os, 2024, https://pypi.org/project/os-sys/ [Ziyaret Tarihi: 2024].

Ozturk, O., Sarıtürk, B. ve Seker, D. Z., 2020, Comparison of fully convolutional networks (FCN) and U-Net for road segmentation from high resolution imageries, *International journal of environment and geoinformatics*, 7(3), 272-279. DOI:10.30897/ijegeo.737993

Önün, F., 2019, Derin öğrenme ile yüksek çözünürlüklü hava görüntülerinde yolların tespit edilmesi, (Yüksek Lisans Tezi), *Fen Bilimleri Enstitüsü*.

Öz, M., 2021, Eye segmentation using deep neural networks, (Yüksek Lisans Tezi), *Akdeniz University*.

Özgür, S. B., 2021, Algoritmalar, Yapay Zeka, Makine Öğrenmesi, Derin Öğrenme ve Uygulamaları: Beşeri Fayda Üretiminin Yazılımlar Tarafından Karşılanması, *Ekonomi ve Yönetim Araştırmaları Dergisi*, 10(1), 1-29.

Öztürk, O., 2023, Deep learning based road segmentation from multi-source and multi-scale data, (Doktora (PhD)), *İstanbul Teknik Üniversitesi*.

Patil, D., Patil, K., Nale, R. ve Chaudhari, S., 2022, Semantic segmentation of satellite images using modified U-Net, *Paper presented at the 2022 IEEE Region 10 Symposium (TENSYMP)*.

Pesaresi, M. ve Gerhardinger, A., 2010, Improved textural built-up presence index for automatic recognition of human settlements in arid regions with scattered vegetation, *IEEE Journal of selected topics in applied earth observations and remote sensing*, 4(1), 16-26. DOI:10.1109/JSTARS.2010.2049478

Pesaresi, M., Gerhardinger, A. ve Kayitakire, F., 2008, A robust built-up area presence index by anisotropic rotation-invariant textural measure, *IEEE Journal of selected topics in applied earth observations and remote sensing*, 1(3), 180-192. DOI:10.1109/JSTARS.2008.2002869

Raju, P., Chaudhary, H. ve Jha, A., 2014, Shadow analysis technique for extraction of building height using high resolution satellite single image and accuracy assessment, *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 40, 1185-1192. DOI:10.5194/isprsarchives-XL-8-1185-2014

Ramachandran, P., Zoph, B. ve Le, Q. V., 2017, Searching for activation functions, *arXiv preprint arXiv:1710.05941*. DOI:10.48550/arXiv.1710.05941
Focus to learn more

Ronneberger, O., Fischer, P. ve Brox, T., 2015, U-net: Convolutional networks for biomedical image segmentation, *Paper presented at the Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*.

Ruder, S., 2016, An overview of gradient descent optimization algorithms, *arXiv preprint arXiv:1609.04747*. DOI:10.48550/arXiv.1609.04747

Saba, F., Valadan Zoej, M. J. ve Mokhtarzade, M., 2016, Optimization of multiresolution segmentation for object-oriented road detection from high-resolution images, *Canadian Journal of Remote Sensing*, 42(2), 75-84. DOI:10.1080/07038992.2016.1160770

Safaei, N., Smadi, O., Safaei, B. ve Masoud, A., 2021, Efficient road crack detection based on an adaptive pixel-level segmentation algorithm, *Transportation Research Record*, 2675(9), 370-381. DOI:10.1177/036119812110022

Saralioglu, E. ve Gungor, O., 2022, Semantic segmentation of land cover from high resolution multispectral satellite images by spectral-spatial convolutional neural network, *Geocarto International*, 37(2), 657-677. DOI:10.1080/10106049.2020.1734871

Sariturk, B. ve Seker, D. Z., 2023, Comparison of residual and dense neural network approaches for building extraction from high-resolution aerial images, *Advances in Space Research*, 71(7), 3076-3089. DOI:10.1016/j.asr.2022.05.010

Schubert, H., van de Gronde, J. J. ve Roerdink, J. B., 2016, Efficient computation of greyscale path openings, *Mathematical Morphology-Theory and Applications*, 1(1). DOI:10.1515/mathm-2016-0010

Seferbekov, S., Iglovikov, V., Buslaev, A. ve Shvets, A., 2018, Feature pyramid network for multi-class land segmentation, *Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition workshops*.

She, Y., 2022, Building instance segmentation in high-resolution remote sensing images based on multi-task learning, *Paper presented at the 2022 IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA)*.

Sherrah, J., 2016, Fully convolutional networks for dense semantic labelling of high-resolution aerial imagery, *arXiv preprint arXiv:1606.02585*. DOI:10.48550/arXiv.1606.02585

Siddiqui, F. U., Awrangjeb, M., Teng, S. W. ve Lu, G., 2016, A new building mask using the gradient of heights for automatic building extraction, *Paper presented at the 2016 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*.

Srivastava, V., Avudaiammal, R. ve V George, S., 2024, Investigations on extraction of buildings from RS imagery using deep learning models, *International journal of remote sensing*, 45(1), 68-100. DOI:10.1080/01431161.2023.2292016

Sutskever, I., Martens, J., Dahl, G. ve Hinton, G., 2013, On the importance of initialization and momentum in deep learning, *Paper presented at the International conference on machine learning*.

Şennur, Ö., 2022, Derin öğrenme ile araç tespiti: Yıldız Teknik Üniversitesi kampüs otopark alanları örneği (Yüksek Lisans), *Yıldız Teknik Üniversitesi*.

Tan, Q., Wang, J. ve Aldred, D. A., 2008, Road vehicle detection and classification from very-high-resolution color digital orthoimagery based on object-oriented method, *Paper presented at the IGARSS 2008-2008 IEEE International Geoscience and Remote Sensing Symposium*.

Tao, C., Tan, Y., Cai, H., Bo, D. ve Tian, J., 2010, Object-oriented method of hierarchical urban building extraction from high-resolution remote-sensing imagery, *Acta Geodaetica et Cartographica Sinica*, 39(1), 39-45.

Tao, Y., Tian, L., Wang, C., Dai, W. ve Xu, Y., 2022, A fine construction method of urban road DEM considering road morphological characteristics, *Scientific reports*, 12(1), 14958. DOI:10.1038/s41598-022-19349-4

Tensorflow, 2024, https://www.tensorflow.org/?hl=tr [Ziyaret Tarihi: 2024].

Tian, J., Cui, S. ve Reinartz, P., 2013, Building change detection based on satellite stereo imagery and digital surface models, *IEEE Transactions on Geoscience and Remote Sensing*, 52(1), 406-417. DOI:10.1109/TGRS.2013.2240692

Tieleman, T. ve Hinton, G., 2012, Rmsprop: Divide the gradient by a running average of its recent magnitude. coursera: Neural networks for machine learning, *COURSERA Neural Networks Mach. Learn*, 17.

Tieleman, T. ve Hinton, G., 2018, Rmsprop: Divide the Gradient by a Running Average of Its Recent Magnitude, 2012, *Coursera. Available online: https://es. coursera. org/learn/neural-networks/lecture/YQHki/rmsprop-divide-the-gradient-by-a-running-average-of-its-recent-magnitude (accessed on 20 April 2018)*.

Tondewad, M. P. S. ve Dale, M. M. P., 2020, Remote sensing image registration methodology: Review and discussion, *Procedia Computer Science*, 171, 2390-2399. DOI:10.1016/j.procs.2020.04.259

Tun, S. W. ve Tun, K. M. M., 2019, Extraction of Buildings in Remote Sensing Imagery with Deep Belief Network, *Paper presented at the 2019 International Conference on Advanced Information Technologies (ICAIT)*.

Turker, M. ve Koc-San, D., 2015, Building extraction from high-resolution optical spaceborne images using the integration of support vector machine (SVM) classification, Hough transformation and perceptual grouping, *International Journal of Applied Earth Observation and Geoinformation*, 34, 58-69. DOI:10.1016/j.jag.2014.06.016

Uray, F., 2022, Derin Öğrenme Tekniklerini Kullanarak Hava Lidar Nokta Bulutlarinin Siniflandirilmasi, (Doktora Tezi (PhD)), *Necmettin Erbakan University (Turkey)*.

URL, Keras Optimizasyonları, https://keras.io/api/optimizers/ [Ziyaret Tarihi: 05 Nisan 2024].

Ünal, A. ve Yıldız, F., 2021, Göktürk-1 uydu görüntülerinin pankeskinleştirme performansının incelenmesi, *Geomatik*, 6(2), 148-164. DOI:10.29128/geomatik.731816

Üzümlü, Y., 2019, Yüksek yoğunluklu kentsel alanlarda çok yüksek çözünürlüklü gerçek ortofotolardan kural tabanlı sınıflandırma yöntemi ile bina çıkarımı, (Yüksek Lisans Tezi), *Fen Bilimleri Enstitüsü*.

Vakalopoulou, M., Karantzalos, K., Komodakis, N. ve Paragios, N., 2015, Building detection in very high resolution multispectral data with deep learning features, *Paper presented at the 2015 IEEE international geoscience and remote sensing symposium (IGARSS)*.

Van de Sande, D., Van Genderen, M. E., Smit, J. M., Huiskens, J., Visser, J. J., Veen, R. E., van Unen, E., Hilgers, O., Gommers, D. ve van Bommel, J., 2022, Developing, implementing and governing artificial intelligence in medicine: a step-by-step approach to prevent an artificial intelligence winter, *BMJ health & care informatics*, 29(1). DOI:10.1136/bmjhci-2021-100495

Vosselman, G. ve De Knecht, J., 1995, Road tracing by profile matching and Kaiman filtering, *Paper presented at the Automatic extraction of man-made objects from aerial and space images*.

Wagner, F. H., Dalagnol, R., Tarabalka, Y., Segantine, T. Y., Thomé, R. ve Hirye, M. C., 2020, U-net-id, an instance segmentation model for building extraction from

satellite images—case study in the joanópolis city, brazil, *Remote Sensing*, 12(10), 1544. DOI:10.3390/rs12101544

Wang, B. ve Ye, Q., 2020, Stochastic gradient descent with nonlinear conjugate gradient-style adaptive momentum, *arXiv preprint arXiv:2012.02188*. DOI:10.48550/arXiv.2012.02188

Wang, E., Qi, K., Li, X. ve Peng, L., 2019, Semantic segmentation of remote sensing image based on neural network, *Acta Optica Sinica*, 39(12), 93-104.

Wang, S., Li, X., Lin, L., Lu, H., Jiang, Y., Zhang, N., Wang, W., Yue, J. ve Li, Z., 2024, A Single Data Extraction Algorithm for Oblique Photographic Data Based on the U-Net, *Remote Sensing*, 16(6), 979. DOI:10.3390/rs16060979

Wang, S., Mu, X., Yang, D., He, H. ve Zhao, P., 2020, Attention guided encoder-decoder network with multi-scale context aggregation for land cover segmentation, *IEEE Access*, 8, 215299-215309.

Wang, W., Xie, E., Song, X., Zang, Y., Wang, W., Lu, T., Yu, G. ve Shen, C., 2019, Efficient and accurate arbitrary-shaped text detection with pixel aggregation network, *Paper presented at the Proceedings of the IEEE/CVF international conference on computer vision*.

Wang, Y., Wang, Y., Da, Y., Liu, X., Li, J. ve Huang, J., 2011, An object-oriented method for road damage detection from high resolution remote sensing images, *Paper presented at the 2011 19th International Conference on Geoinformatics*.

Wei, Y., Zhao, Z. ve Song, J., 2004, Urban building extraction from high-resolution satellite panchromatic image using clustering and edge detection, *Paper presented at the IGARSS 2004. 2004 IEEE international geoscience and remote sensing symposium*.

Wu, G., Shao, X., Guo, Z., Chen, Q., Yuan, W., Shi, X., Xu, Y. ve Shibasaki, R., 2018, Automatic building segmentation of aerial imagery using multi-constraint fully convolutional networks, *Remote Sensing*, 10(3), 407. DOI:10.3390/rs10030407

Wu, H., Cheng, Z., Shi, W., Miao, Z. ve Xu, C., 2014, An object-based image analysis for building seismic vulnerability assessment using high-resolution remote sensing imagery, *Natural Hazards*, 71, 151-174. DOI:10.1007/s11069-013-0905-6

Wu, M., Zhang, C., Liu, J., Zhou, L. ve Li, X., 2019, Towards accurate high resolution satellite image semantic segmentation, *IEEE Access*, 7, 55609-55619. DOI:10.1109/ACCESS.2019.2913442

Xiang, S., Xie, Q. ve Wang, M., 2021, Semantic segmentation for remote sensing images based on adaptive feature selection network, *IEEE Geoscience and Remote Sensing Letters*, 19, 1-5. DOI:10.1109/LGRS.2021.3049125

Xu, L., Liu, Y., Yang, P., Chen, H., Zhang, H., Wang, D. ve Zhang, X., 2021, HA U-Net: Improved model for building extraction from high resolution remote sensing imagery, *IEEE Access*, 9, 101972-101984. DOI:10.1109/ACCESS.2021.3097630

Yan, X., Tang, H., Sun, S., Ma, H., Kong, D. ve Xie, X., 2022, After-unet: Axial fusion transformer unet for medical image segmentation, *Paper presented at the Proceedings of the IEEE/CVF winter conference on applications of computer vision*.

Yang, B., Huang, R., Li, J., Tian, M., Dai, W. ve Zhong, R., 2016, Automated reconstruction of building LoDs from airborne LiDAR point clouds using an improved morphological scale space, *Remote Sensing*, 9(1), 14. DOI:10.3390/rs9010014

Yang, H. L., Lunga, D. ve Yuan, J., 2017, Toward country scale building detection with convolutional neural network using aerial images, *Paper presented at the 2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*.

Yang, H. L., Yuan, J., Lunga, D., Laverdiere, M., Rose, A. ve Bhaduri, B., 2018, Building extraction at scale using convolutional neural network: Mapping of the united states, *IEEE Journal of selected topics in applied earth observations and remote sensing*, 11(8), 2600-2614. DOI:10.1109/JSTARS.2018.2835377

Yiğit, A. Y. ve Uysal, M., 2019, Nesne Tabanlı Sınıflandırma Yaklaşımı Kullanılarak Yolların Tespiti, *Türkiye Fotogrametri Dergisi*, 1(1), 17-24.

Yılmaz, E. Ö., 2020, Uzaktan algılanmış verilerin derin öğrenme yöntemiyle sınıflandırılması, (Yüksek Lisans), *Gebze Teknik Üniversitesi*.

Yılmaz, E. Ö. ve Kavzoğlu, T., 2021, Derin Öğrenmenin Temel Prensipleri ve Uzaktan Algılama Alanındaki Uygulamaları, *Harita Dergisi*, 87(166), 25-43.

Yu, L., Ma, F., Jayasuriya, A., Sigelle, M. ve Perreau, S., 2007, A new contour detection approach in mammogram using rational wavelet filtering and MRF smoothing, *Paper presented at the 9th Biennial Conference of the Australian Pattern Recognition Society on Digital Image Computing Techniques and Applications (DICTA 2007)*.

Yuan, D., Fan, N. ve He, Z., 2020, Learning target-focusing convolutional regression model for visual object tracking, *Knowledge-Based Systems*, 194, 105526. DOI:10.1016/j.knosys.2020.105526

Zeiler, M. D., 2012, Adadelta: an adaptive learning rate method, *arXiv preprint arXiv:1212.5701*. DOI:10.48550/arXiv.1212.5701

Zeiler, M. D., Ranzato, M., Monga, R., Mao, M., Yang, K., Le, Q. V., Nguyen, P., Senior, A., Vanhoucke, V. ve Dean, J., 2013, On rectified linear units for speech processing, *Paper presented at the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing*.

Zhang, J., Du, J., Liu, H., Hou, X., Zhao, Y. ve Ding, M., 2019, LU-NET: An improved U-Net for ventricular segmentation, *IEEE Access*, 7, 92539-92546. DOI:10.1109/ACCESS.2019.2925060

Zhang, J., Lin, X., Liu, Z. ve Shen, J., 2011, Semi-automatic road tracking by template matching and distance transformation in urban areas, *International journal of remote sensing*, 32(23), 8331-8347. DOI:10.1080/01431161.2010.540587

Zhang, N., Lei, D. ve Zhao, J., 2018, An improved Adagrad gradient descent optimization algorithm, *Paper presented at the 2018 Chinese Automation Congress (CAC)*.

Zhang, P., Gong, M., Su, L., Liu, J. ve Li, Z., 2016, Change detection based on deep feature representation and mapping transformation for multi-spatial-resolution remote sensing images, *ISPRS Journal of photogrammetry and remote sensing*, 116, 24-41. DOI:10.1016/j.isprsjprs.2016.02.013

Zhao, H., Shi, J., Qi, X., Wang, X. ve Jia, J., 2017, Pyramid scene parsing network, *Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition*.

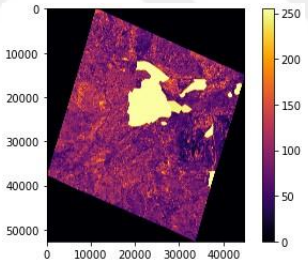
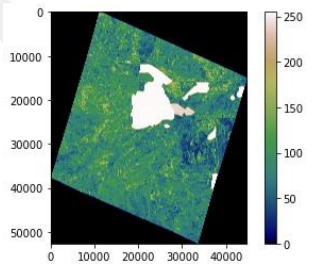
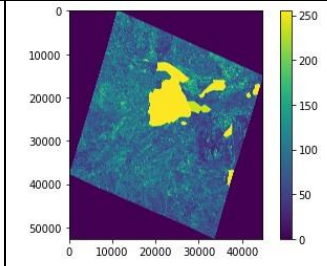
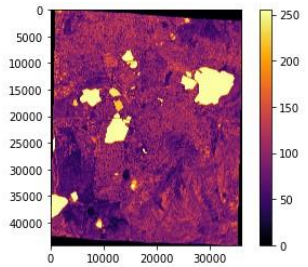
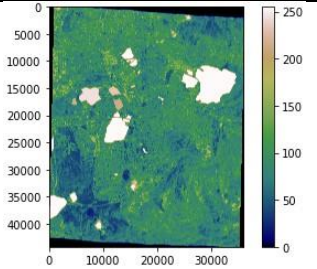
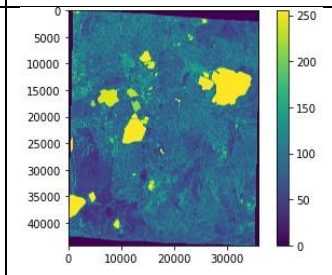
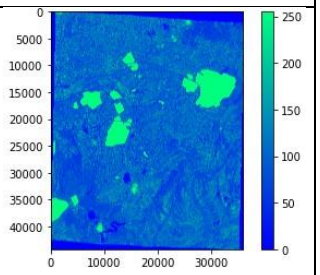
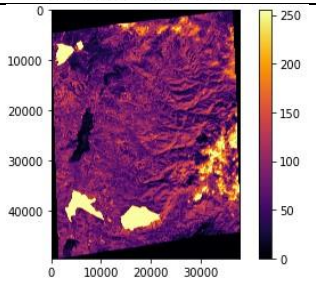
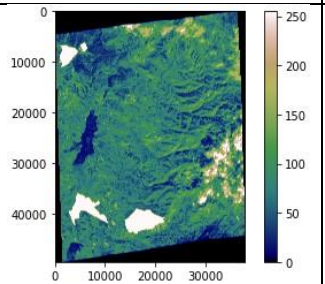
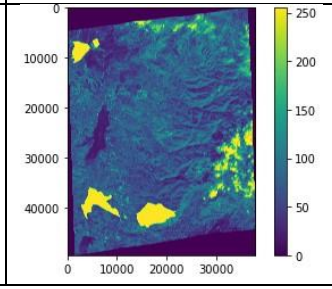
Zheng, L., Ai, P. ve Wu, Y., 2020, Building recognition of UAV remote sensing images by deep learning, *Paper presented at the IGARSS 2020-2020 IEEE International Geoscience and Remote Sensing Symposium*.

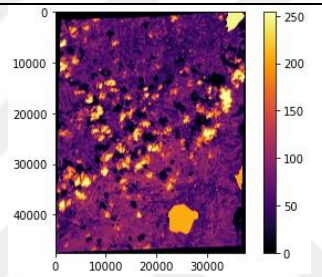
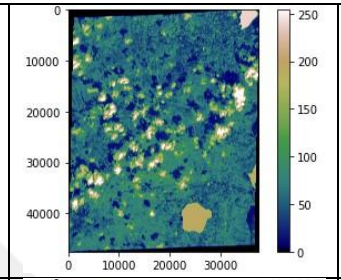
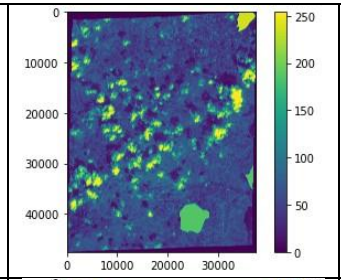
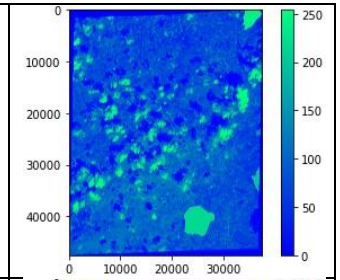
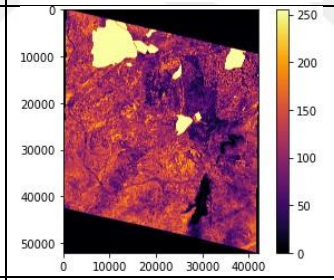
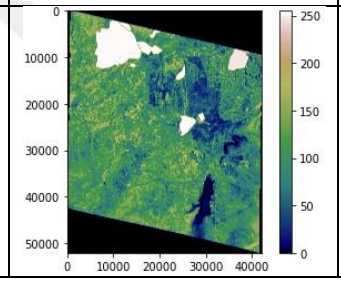
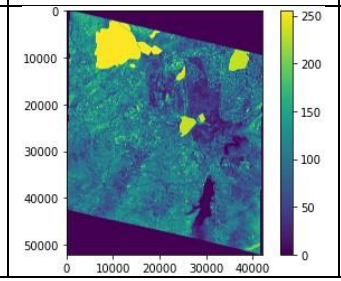
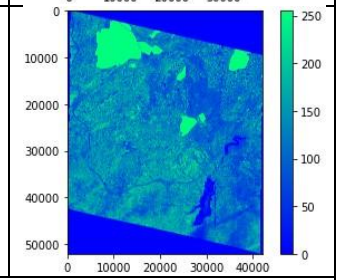
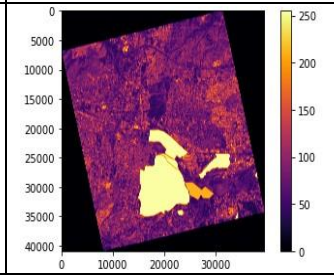
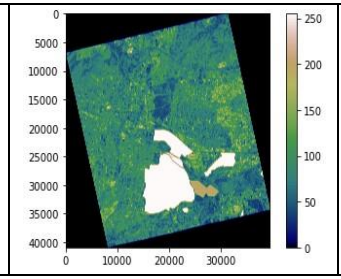
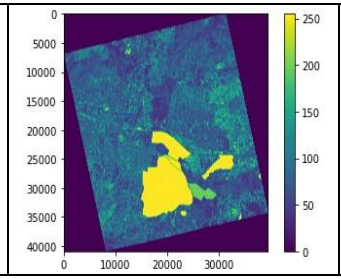
Zhou, Z., Siddiquee, M. M. R., Tajbakhsh, N. ve Liang, J., 2019, Unet++: Redesigning skip connections to exploit multiscale features in image segmentation, *IEEE transactions on medical imaging*, 39(6), 1856-1867. DOI:10.1109/TMI.2019.2959609

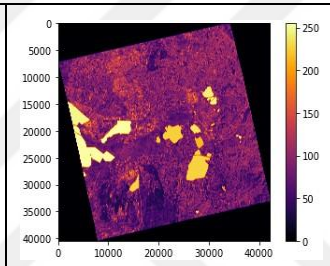
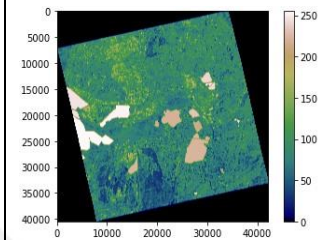
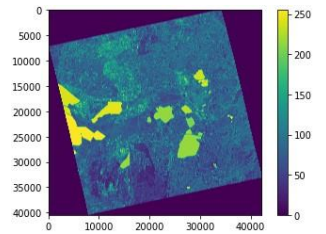
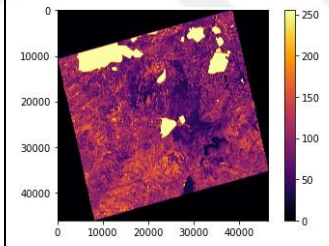
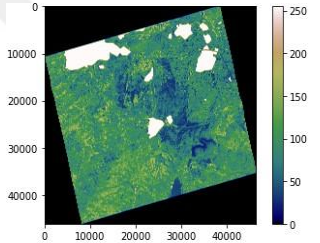
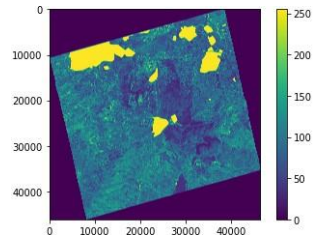
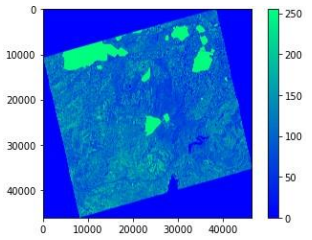
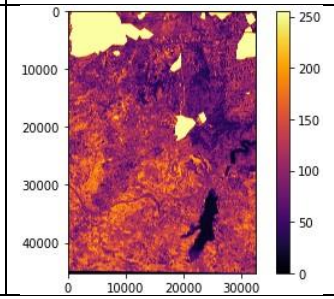
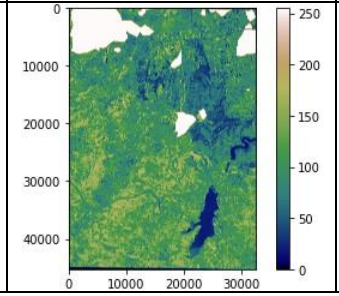
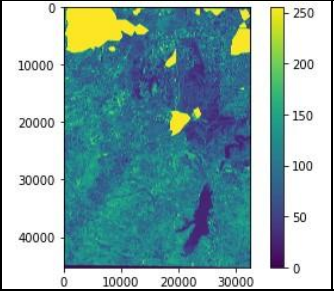
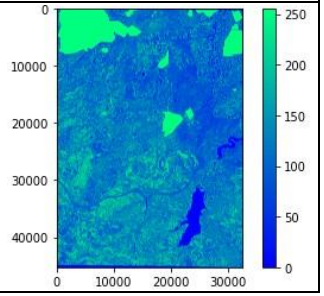
Zhu, Q., Liao, C., Hu, H., Mei, X. ve Li, H., 2020, MAP-Net: Multiple attending path neural network for building footprint extraction from remote sensed imagery, *IEEE Transactions on Geoscience and Remote Sensing*, 59(7), 6169-6181. DOI:10.1109/TGRS.2020.3026051

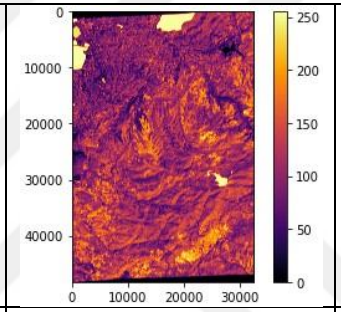
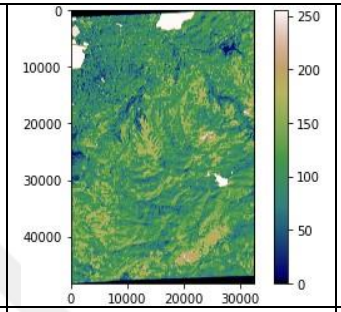
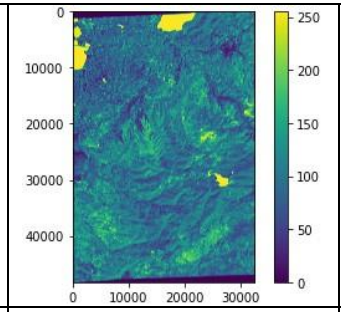
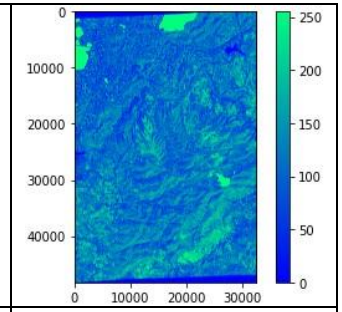
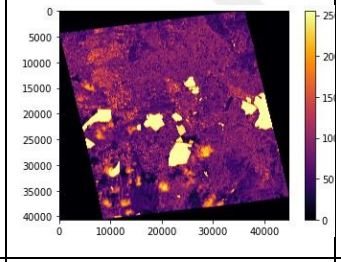
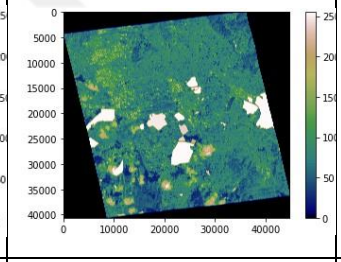
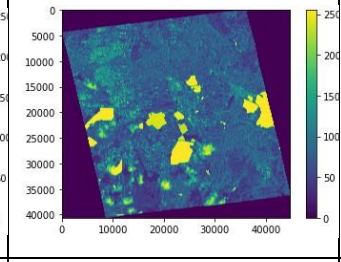
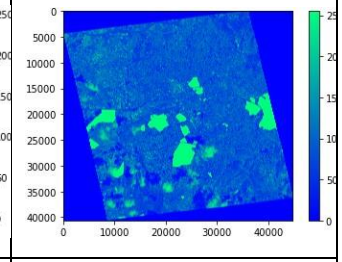
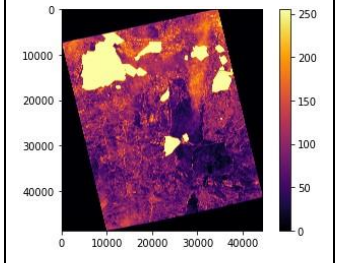
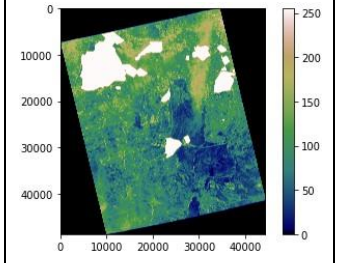
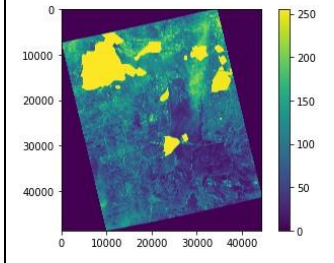
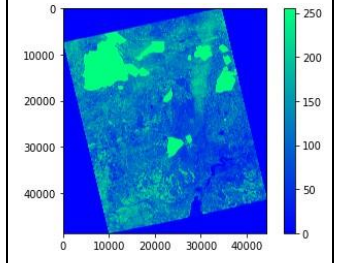
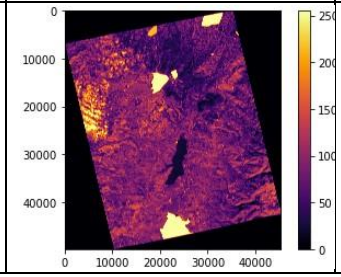
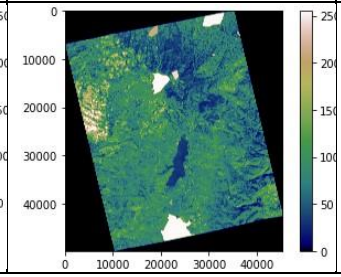
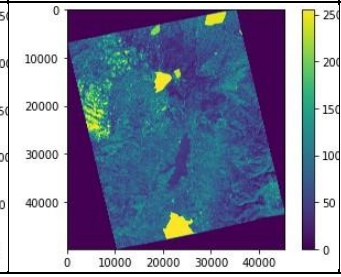
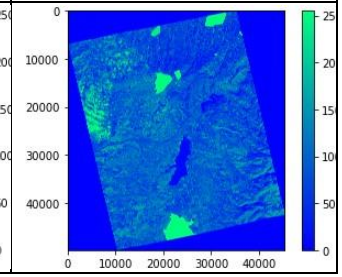
EKLER

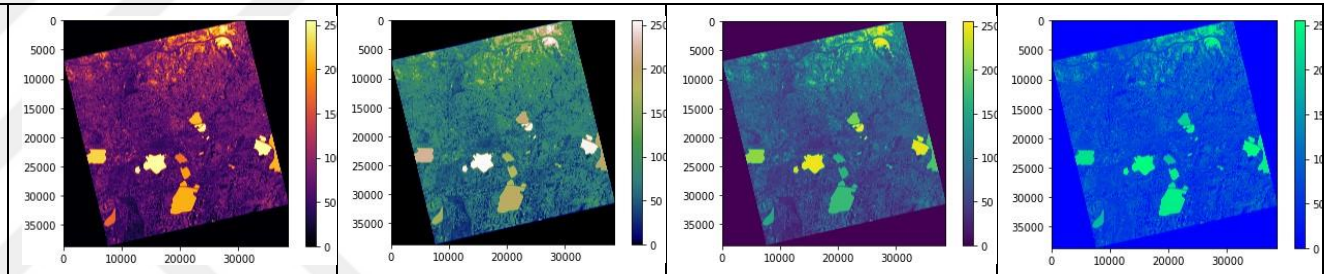
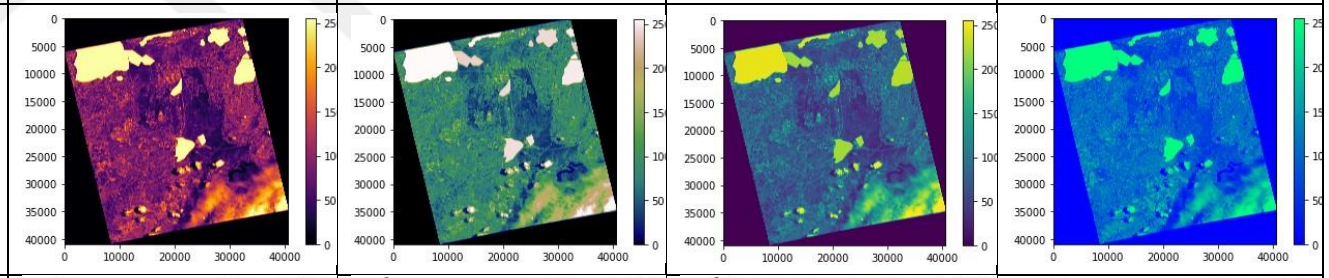
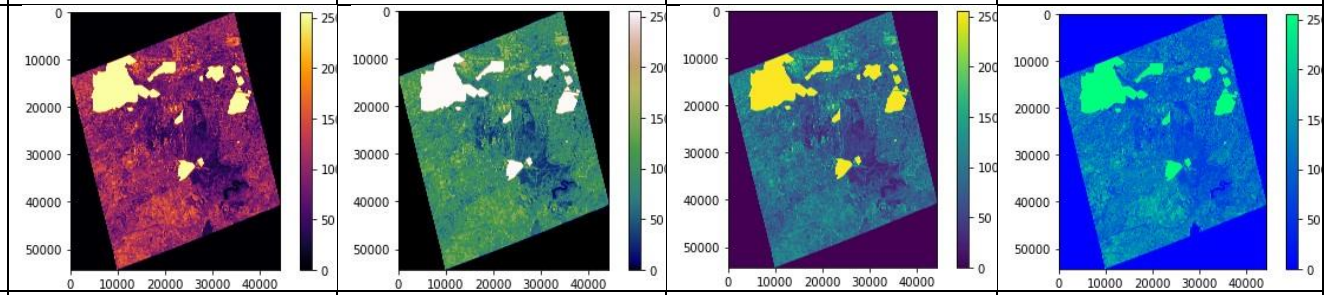
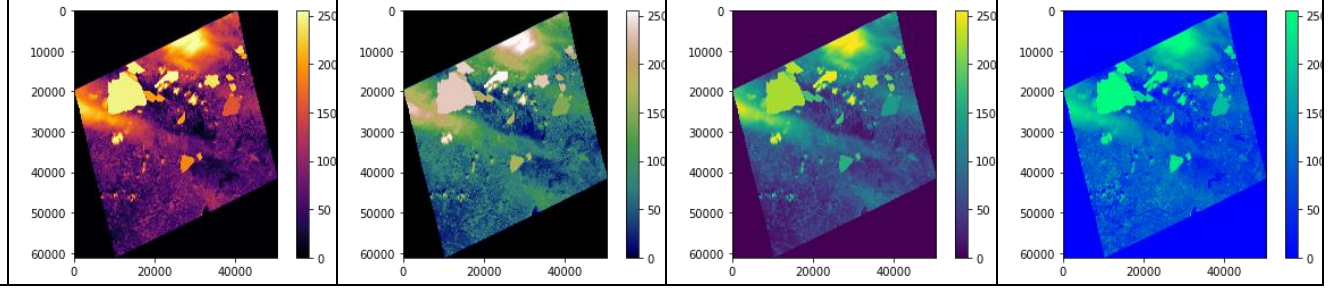
EK-1 Temin edilen Göktürk-1 uydu görüntüsüne ait bilgiler

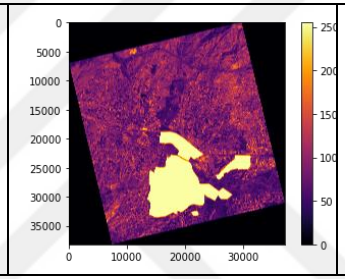
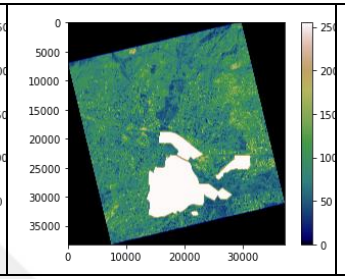
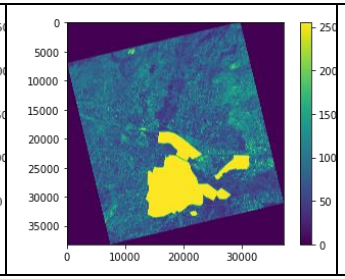
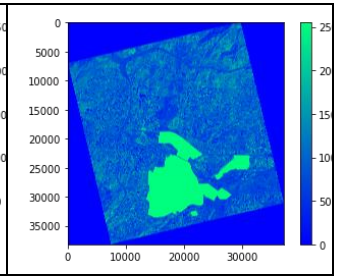
Yıl	Görüntü Adı	En Boy	Satır Sütun	Koordinatlar	Kırmızı Bant	Yeşil Bant	Mavi Bant	Kızılötesi Bant
2018	GKT1_L3B_30_ETIMESGUT_0HGM050CBUG_20180706_UTM	En: 22323.0 Boy: 26320.5	Satır: 52641 Sütun: 44646	Sol üst köşe koordinatları: (461296.0, 4429677.5) Sağ alt köşe koordinatları: (487616.0, 4407355.0)				YOK
2018	GKT1_L3B_31_ANKARA_0HGM050CBUG_20180719_UTM	En: 17990.0, Boy: 22142.5	Satır: 44285 Sütun: 35980	Sol üst köşe koordinatları: (479628.0, 4428684.0) Sağ alt köşe koordinatları: (501770.0, 4410694.5)				
2018	GKT1_L3B_34_GOLBASI_0HGM050CBUG_20181114_UTM	En: 18893.5 Boy: 24764.5	Satır: 49529 Sütun: 37787	Sol üst köşe koordinatları: (479020.0, 4414685.5) Sağ alt köşe koordinatları: (503784.0, 4395792.5)				YOK

2018	GKT1_L3B_35_GOLBASI_0HGM050CBUG 20180801 UTM	En: 18749.0 Boy: 23759.0	Satır: 47518 Sütun: 37498	Sol üst köşe koordinatları: (462659.0, 4411083.5) Sağ alt köşe koordinatları: (486417.5, 4392335.0)					
2018	GKT1_L3B_36_GOLBASI_0HGM050CBUG 20181003 UTM	En: 20971.5 Boy: 26059.0	Satır: 52118 Sütun: 41943	Sol üst köşe koordinatları: (467370.0, 4422585.5) Sağ alt köşe koordinatları: (493428.5, 4401614.5)					
2019	GKT1_L3B_15_ETIMESGUT_0HGM050CBUG 20190830 UTM	En: 45960.5 Boy: 132860.0	Satır: 265720, Sütun: 91921	Sol üst köşe koordinatları: (441400.0, 4510736.0) Sağ alt köşe koordinatları: (574259.5, 4464776.0)	4 bantlı, boyutları 265720×91921 ve uint8 veri türüne sahip bir dizi için 91,0 GiB ayrılamıyor bellek hatası verdiği için görüntüleri üretilememiştir.				
2019	GKT1_L3B_18_ETIMESGUT_0HGM050CBUG 20190607 UTM	En: 19722.5 Boy: 20443.5	Satır: 40887 Sütun: 39445	Sol üst köşe koordinatları: (462899.0, 4433854.5) Sağ alt köşe koordinatları: (483342.0, 4414132.5)				Yok	

2019	GKT1_L3B_22_ANKARA_0HGM050CBUG 20190310 UTM	En: 21049.5 Boy: 20234.0	Satır: 40468, Sütun: 42099	Sol üst köşe koordinatları: (472024.0, 4430761.5) Sağ alt köşe koordinatları: (492257.5, 4409712.5)				Yok	
2019	GKT1_L3B_30_ANKARA_0HGM050CBUG 20190923 UTM	En: 23170.5 Boy: 23060.5	Satır: 46121, Sütun: 46341	Sol üst köşe koordinatları: (468207.5, 4423132.5) Sağ alt köşe koordinatları: (491267.5, 4399962.5)					
2020	GKT1_L3B_15_ANKARA_0HGM050CBUG 20200916 UTM	En: 16311.5, Boy: 22633.5	Satır: 45267, Sütun: 32623	Sol üst köşe koordinatları: (470246.0, 4420325.5) Sağ alt köşe koordinatları: (492879.0, 4404014.5)					

2020	GKT1_L3B_17_ANKARA_0HGM050CBUG 20200916 UTM	En: 16212.5 Boy: 24124.5	Satır: 48249, Sütun: 32425	Sol üst köşe koordinatları: (485674.5, 4421531.0) Sağ alt köşe koordinatları: (509798.5, 4405319.0)				
2020	GKT1_L3B_27_ANKARA_0HGM050CBUG 20200524 UTM	En: 22370.5 Boy: 20312.5	Satır: 40625, Sütun: 44741	Sol üst köşe koordinatları: (474161.0, 4431265.0) Sağ alt köşe koordinatları: (494473.0, 4408895.0)				
2020	GKT1_L3B_31_ANKARA_0HGM050CBUG 20200508 UTM	En: 22140.5 Boy: 24365.5	Satır: 48731, Sütun: 44281	Sol üst köşe koordinatları: (468228.0, 4425270.0) Sağ alt köşe koordinatları: (492593.0, 4403130.0)				
2020	GKT1_L3B_33_GOLBASI_0HGM050CBUG 20201225 UTM	En: 22593.0 Boy: 24932.5	Satır: 49865, Sütun: 45186	Sol üst köşe koordinatları: (470430.0, 4417799.5) Sağ alt köşe koordinatları: (495362.0, 4395207.0)				

2021	GKT1_L3B_11_ANKARA_0HGM050CBUG 20210123 UTM	En: 19207.5 Boy: 19329.5	Satır: 38659, Sütun: 38415	Sol üst köşe koordinatları: (475605.5, 4432736.0) Sağ alt köşe koordinatları: (494934.5, 4413529.0)	
2021	GKT1_L3B_20_ANKARA_0HGM050CBUG 20210624 UTM	En: 20258.0 Boy: 20481.0	Satır: 40962, Sütun: 40516	Sol üst köşe koordinatları: (469467.5, 4422149.0) Sağ alt köşe koordinatları: (489948.0, 4401891.5)	
2021	GKT1_L3B_32_ANKARA_0HGM050CBUG 20210728 UTM	En: 22146.0 Boy: 27159.5	Satır: 54319, Sütun: 44292	Sol üst köşe koordinatları: (468155.5, 4426918.5) Sağ alt köşe koordinatları: (495314.5, 4404773.0)	
2021	GKT1_L3B_38_ANKARA_0HGM050CBUG 20210621 UTM	En: 25283.5 Boy: 30571.0	Satır: 61142, Sütun: 50567	Sol üst köşe koordinatları: (466313.0, 4429191.0) Sağ alt köşe koordinatları: (496883.5, 4403908.0)	

2021	GKT1_L3B_4_ETIMESGUT_0HGM050CBUG 20210226 UTM	En: 18606.5 Boy: 19096.0	Satır: 38192, Sütun: 37213	Sol üst köşe koordinatları: (463564.5, 4433122.0) Sağ alt köşe koordinatları: (482660.0, 4414516.0)				

EK-2 Göktürk-1 uydu görüntüsüne ait bilgilerinin çıkarılmasına ait kodlar

In [1]:
```
import math
import rasterio
import matplotlib.pyplot as plt

image_file = "G18b2_0HGM036CBHF20210613UTM.tif"
sat_data = rasterio.open(image_file)
```

```
C:\Users\Duygu\anaconda3\lib\site-packages\numpy\_distributor_init.py:30: UserWarning: loaded more than 1 DLL from .libs:
C:\Users\Duygu\anaconda3\lib\site-packages\numpy\.libs\libopenblas.EL2C6PLE4ZYW3ECEVIV3OXXGRN2NRFM2.gfortran-win_amd64.dll
C:\Users\Duygu\anaconda3\lib\site-packages\numpy\.libs\libopenblas.GK7GX5KEQ4F6UYO3P26ULGBQYHGQO7J4.gfortran-win_amd64.dll
  warnings.warn("loaded more than 1 DLL from .libs:"
```

In [2]:
```
width_in_projected_units = sat_data.bounds.right - sat_data.bounds.left
height_in_projected_units = sat_data.bounds.top - sat_data.bounds.bottom

print("Width: {}, Height: {}".format(width_in_projected_units, height_in_projected_units))
```

```
Width: 10602.719999999972, Height: 13938.839999999851
```

In [3]:
```
print("Rows: {}, Columns: {}".format(sat_data.height, sat_data.width))
```

```
Rows: 38719, Columns: 29452
```

In [4]:
```
row_min = 0
col_min = 0


row_max = sat_data.height - 1
col_max = sat_data.width - 1


topleft = sat_data.transform * (row_min, col_min)
botright = sat_data.transform * (row_max, col_max)

print("Top left corner coordinates: {}".format(topleft))
print("Bottom right corner coordinates: {}".format(botright))
```

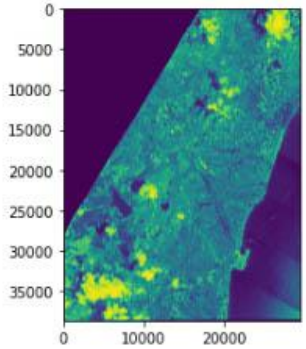
```
Top left corner coordinates: (531532.8, 4538882.52)
Bottom right corner coordinates: (545471.28, 4528280.159999999)
```

In [5]:
```
print(sat_data.count)

print(sat_data.indexes)
```

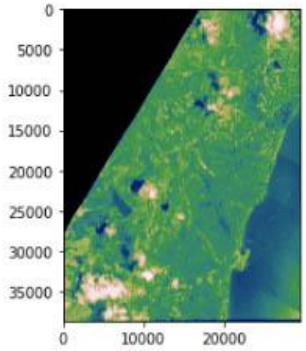
```
4
(1, 2, 3, 4)
```

In [6]:
```
b, g, r, n = sat_data.read()
```

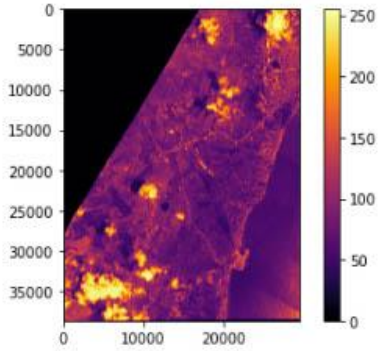
In [7]:
```
fig = plt.imshow(b)
plt.show()
```



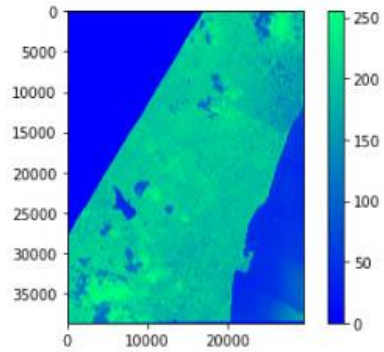
In [8]:
```
fig = plt.imshow(g)
fig.set_cmap('gist_earth')
plt.show()
```



In [9]:
```
fig = plt.imshow(r)
fig.set_cmap('inferno')
plt.colorbar()
plt.show()
```



In [10]:
```
fig = plt.imshow(n)
fig.set_cmap('winter')
plt.colorbar()
plt.show()
```


0
5000
10000
15000
20000
25000
30000
35000
0
10000
20000
250
200
150
100
50
0

In [ ]:

EK-3 Görüntülerin İstenilen Formatta Kırpılması

In [1]:
```
from PIL import Image
import os
import sys
import numpy as np
from natsort import natsorted
import glob

import cv2
import matplotlib.pyplot as plt
```

```
C:\Users\Duygu\anaconda3\lib\site-packages\numpy\_distributor_init.py:30: UserWarning: loaded more than 1 DLL from .libs:
C:\Users\Duygu\anaconda3\lib\site-packages\numpy\.libs\libopenblas.EL2C6PLE4ZYW3ECEVIV3OXXGRN2NRFM2.gfortran-win_amd64.dll
C:\Users\Duygu\anaconda3\lib\site-packages\numpy\.libs\libopenblas.GK7GX5KEQ4F6UYO3P26ULGBQYHGQO7J4.gfortran-win_amd64.dll
  warnings.warn("loaded more than 1 DLL from .libs:"
```

In [2]:
```
#Dosya Okuma
x_train_files = glob.glob('2018_10.tiff')
```

In [3]:
```
#Okunan dosyayı çağırma ve ekrana gösterme
im_image = Image.open(x_train_files[0])

#Görüntüyü diziye dönüştürme
im_list_image = np.asarray(im_image)

#Görselleştirme
plt.imshow(im_list_image)
plt.show()
```

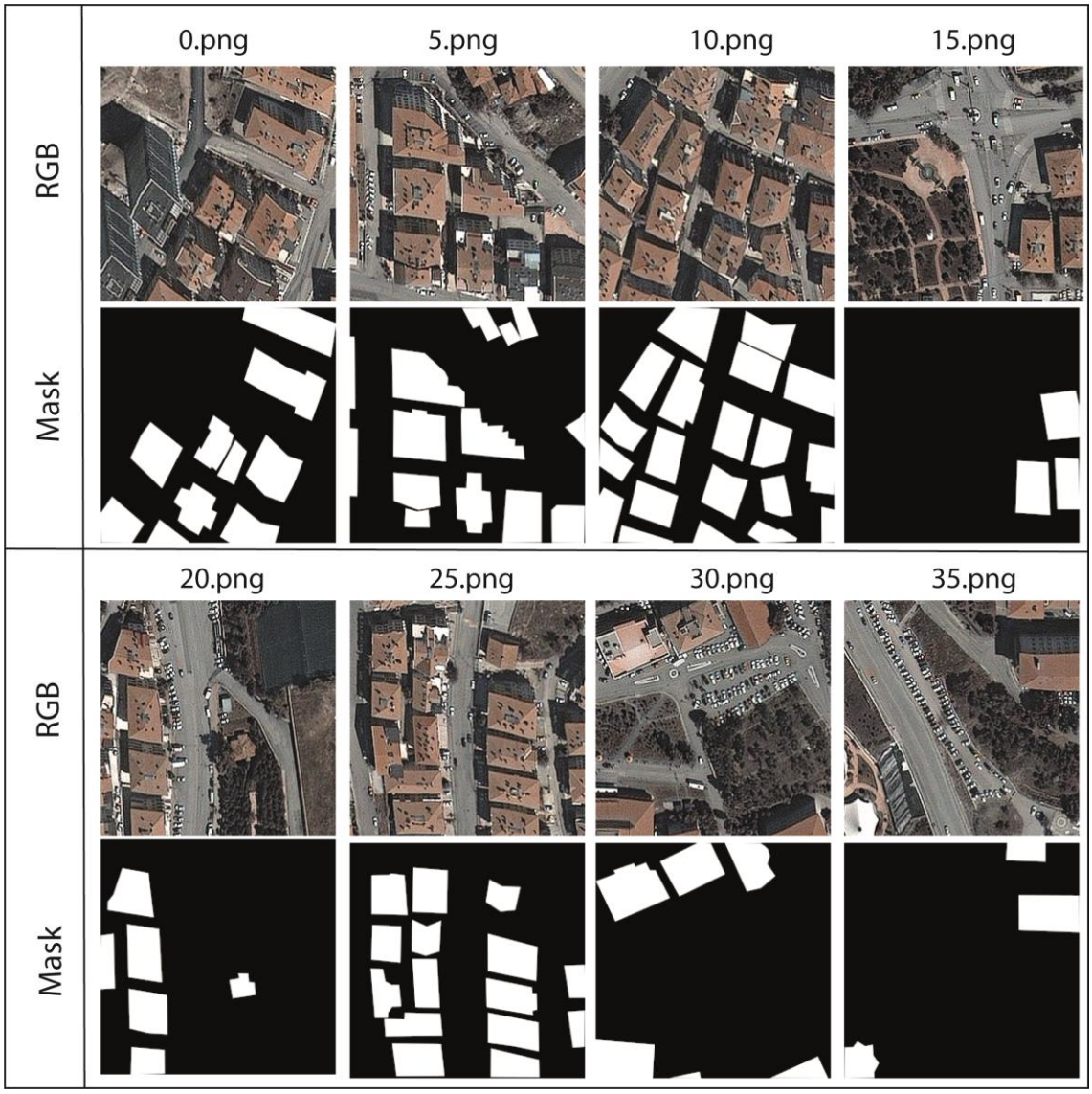
In [4]:
```
#görüntünün boyutlarını kontrol etme
print(im_list_image)
print(len(im_list_image))
print(im_list_image.shape)
```

```
[[[ 57  59  69]
  [130 128 134]
  [145 139 143]
  ...
  [127 129 130]
  [135 136 137]
  [ 94  98 102]]

 [[ 31  31  38]
  [ 83  82  95]
  [ 89  83  94]
  ...
  [161 161 162]
  [165 165 166]
  [101 107 111]]

 [[ 13  13  20]
  [ 30  26  33]
  [ 12   8  13]
  ...
  [133 137 140]
  [133 137 140]
  [ 75  85  91]]

 ...

 [[ 78  83  94]
  [ 25  26  32]
  [ 30  33  40]
  ...
  [ 78  85  81]
  [ 67  75  71]
  [ 89  98  95]]

 [[ 18  19  24]
  [ 20  22  28]
  [ 42  46  55]
  ...
  [158 160 155]
  [ 72  82  75]
  [ 46  52  44]]

 [[ 15  17  21]
  [ 36  39  45]
  [ 45  51  60]
  ...
  [217 216 209]
  [169 170 164]
  [ 53  60  51]]]
1726
(1726, 3167, 3)
```

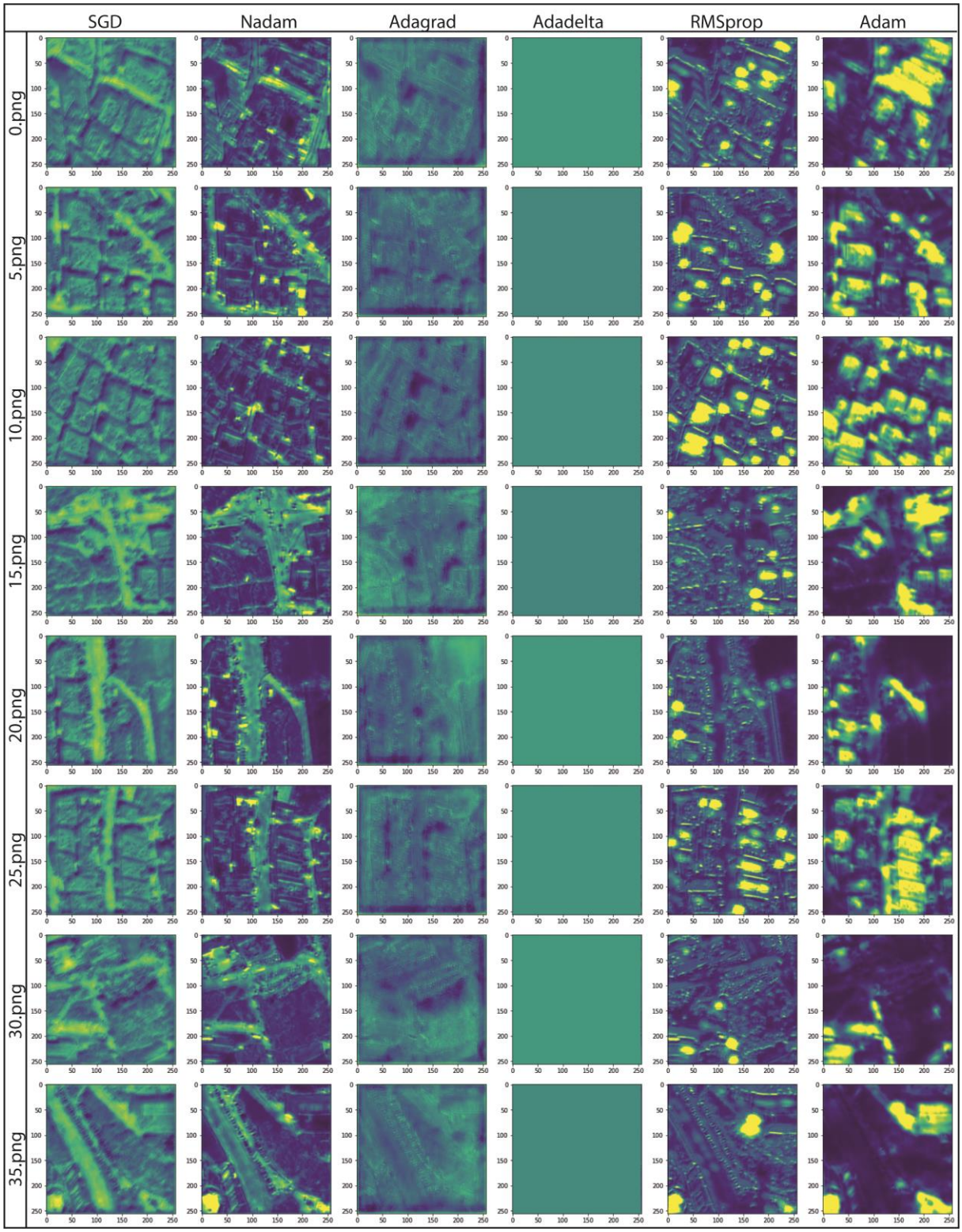
In [5]:
```
#büyük görüntüyü istenilen boyutta bölme
height = 256
width = 256
img_size= len(im_list_image)
```

In [ ]:
```
#Görüntü bölme işleme işlevi
def ImgSplit(im):
    buff = []
    #Dikey bölme numarası
    for h1 in range(int(img_size/height)):
       # Yatay bölünmüş sayı
        for w1 in range(int(img_size/width)):
            w2 = w1 * height
            h2 = h1 * width
            #print(w2, h2, width + w2, height + h2)
            c = im.crop((w2, h2, width + w2, height + h2))
            buff.append(c)
    return buff

#Yakalanan görüntünün bölünmesi
for i in range(len(x_train_files)):
#Resmi yükle
    im=Image.open(x_train_files[i])
    file_name = os.path.splitext(os.path.basename(x_train_files[i]))[0]

    # Görüntü bölme işleminin yürütülmesi
    hi=0
    for ig in ImgSplit(im):
        hi=hi+1
        #Kaydetme hedef klasörünü belirtin
        ig.save("C:/Users/Duygu/Desktop/tik4\tiff_10\2018_10_256/"+ str(file_name) + '_' + str(i) +'_' + str(hi) +".png")
```

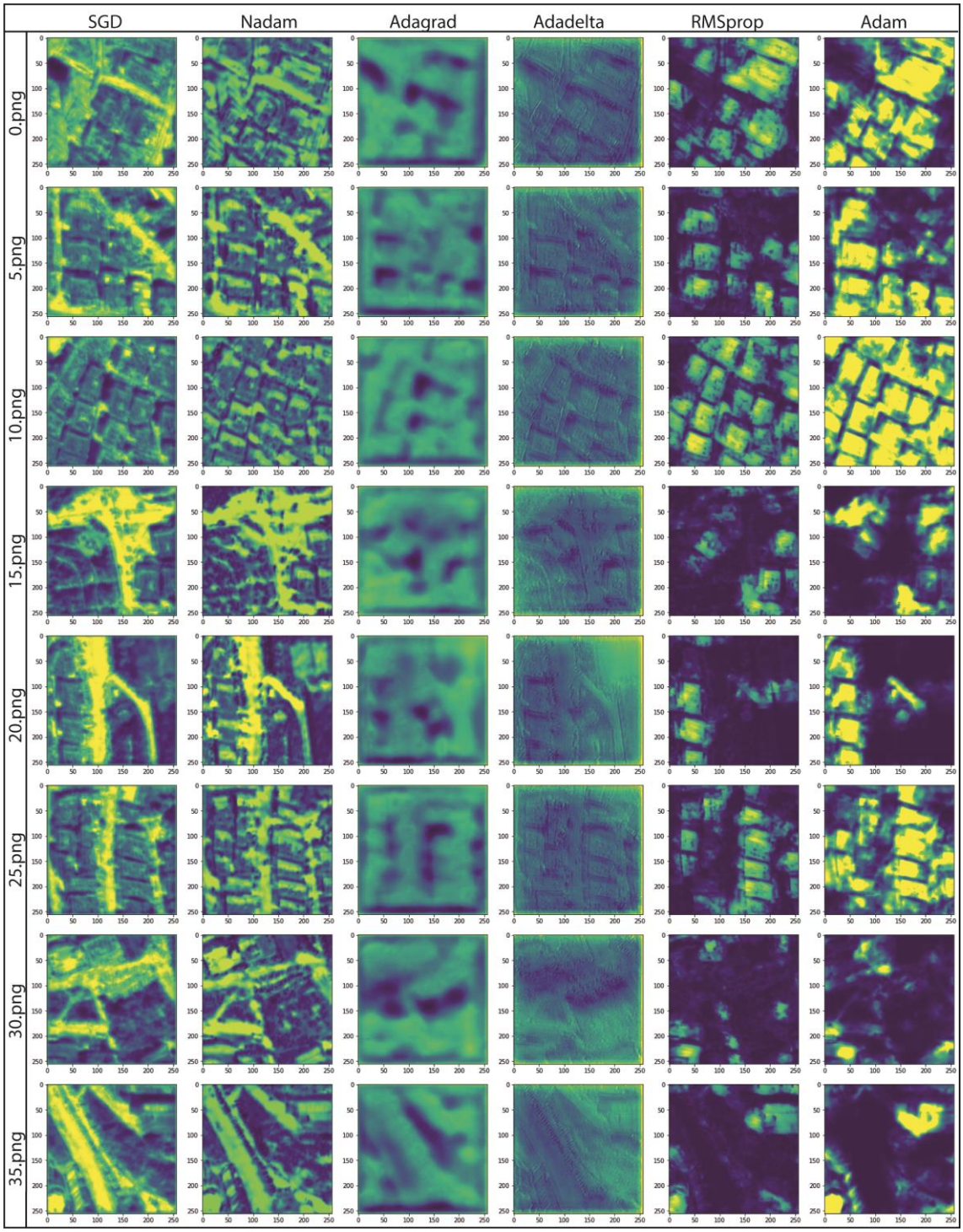
EK-4 Tiff-10 RGB ve Mask görüntüsü



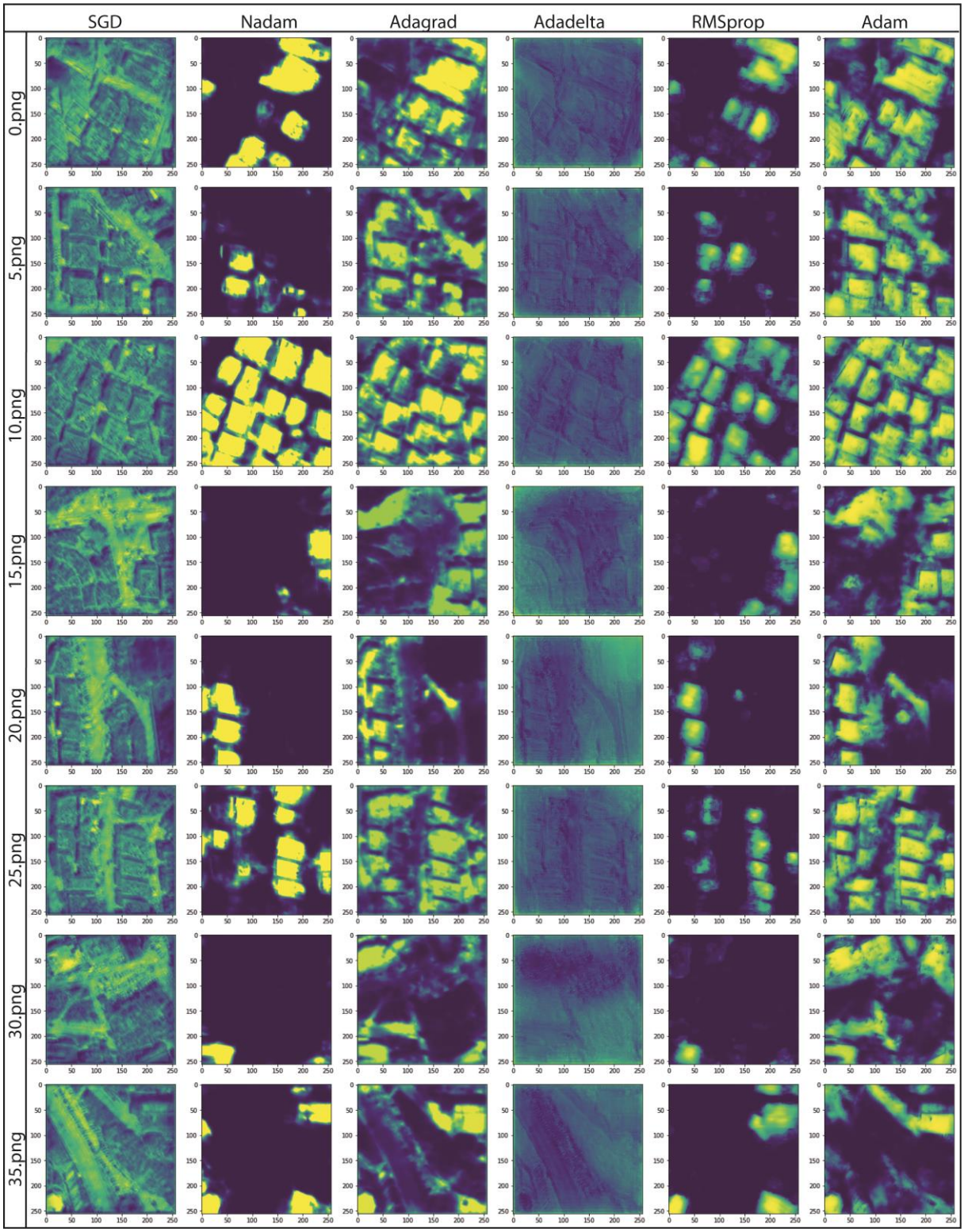
EK-5 Tiff-10 görüntülerinin 10 Epok Sonucunda tahmin edilen görüntüleri



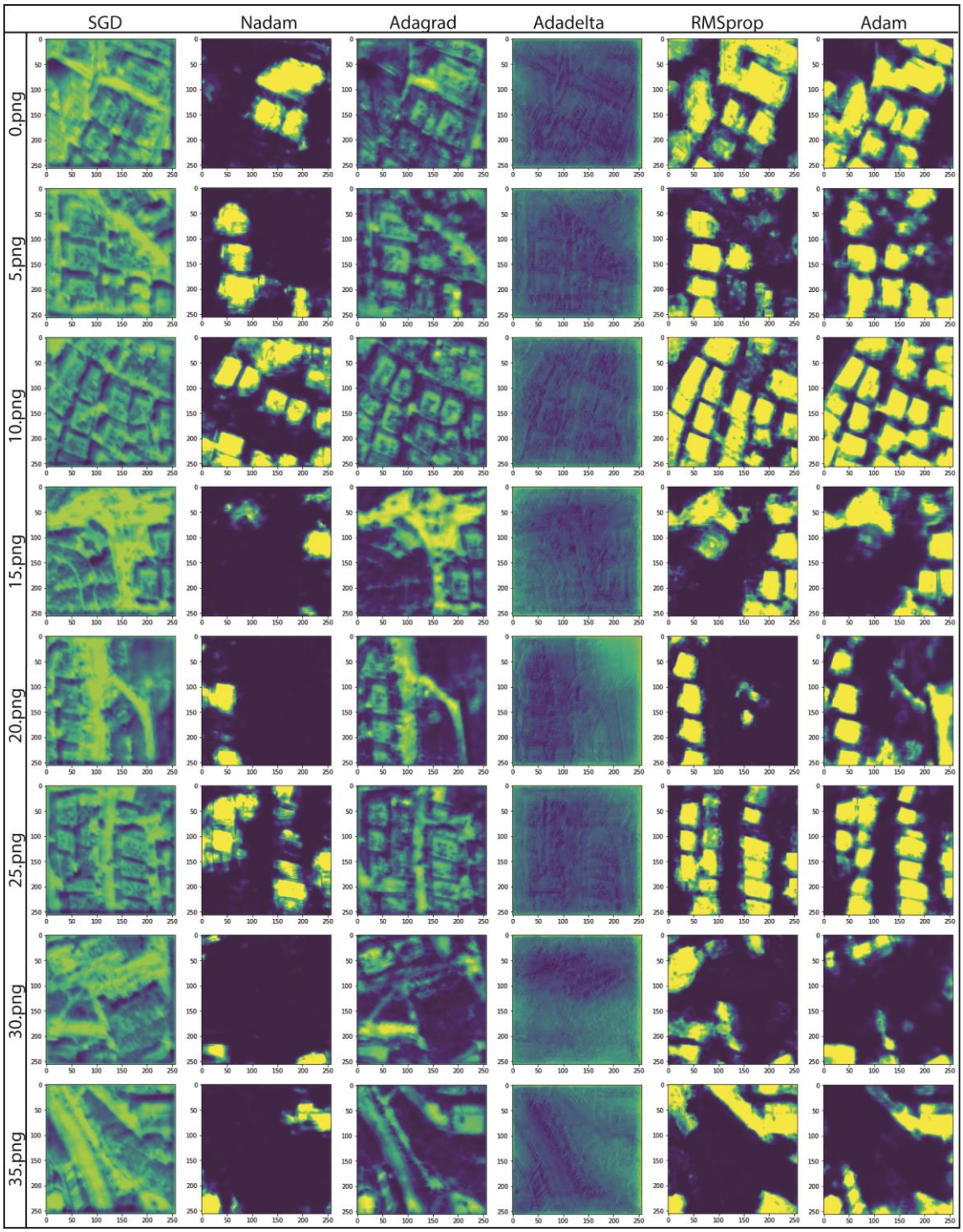
EK-6 Tiff-10 görüntülerinin 25 Epok Sonucunda tahmin edilen görüntüleri



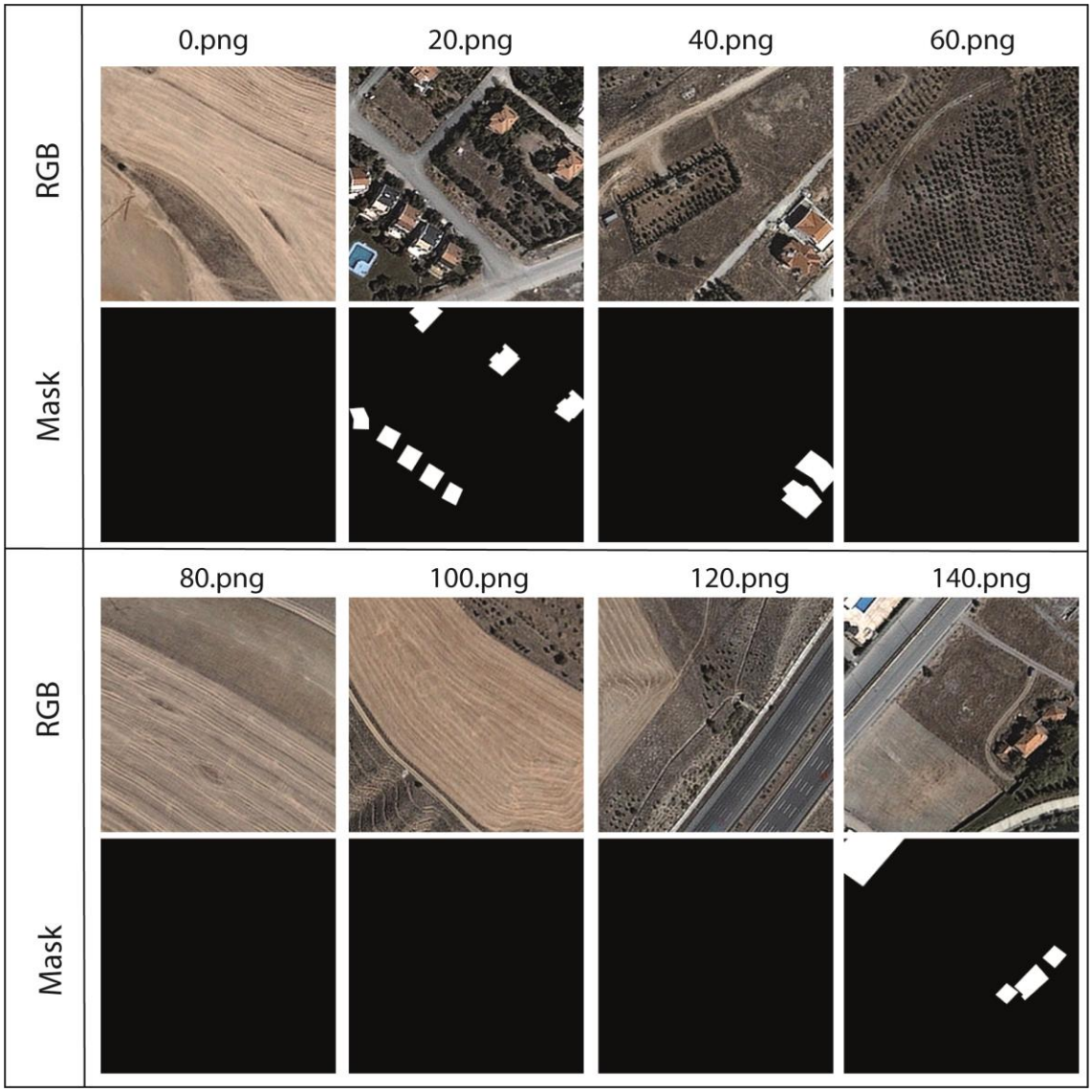
EK-7 Tiff-10 görüntülerinin 50 Epok Sonucunda tahmin edilen görüntüleri



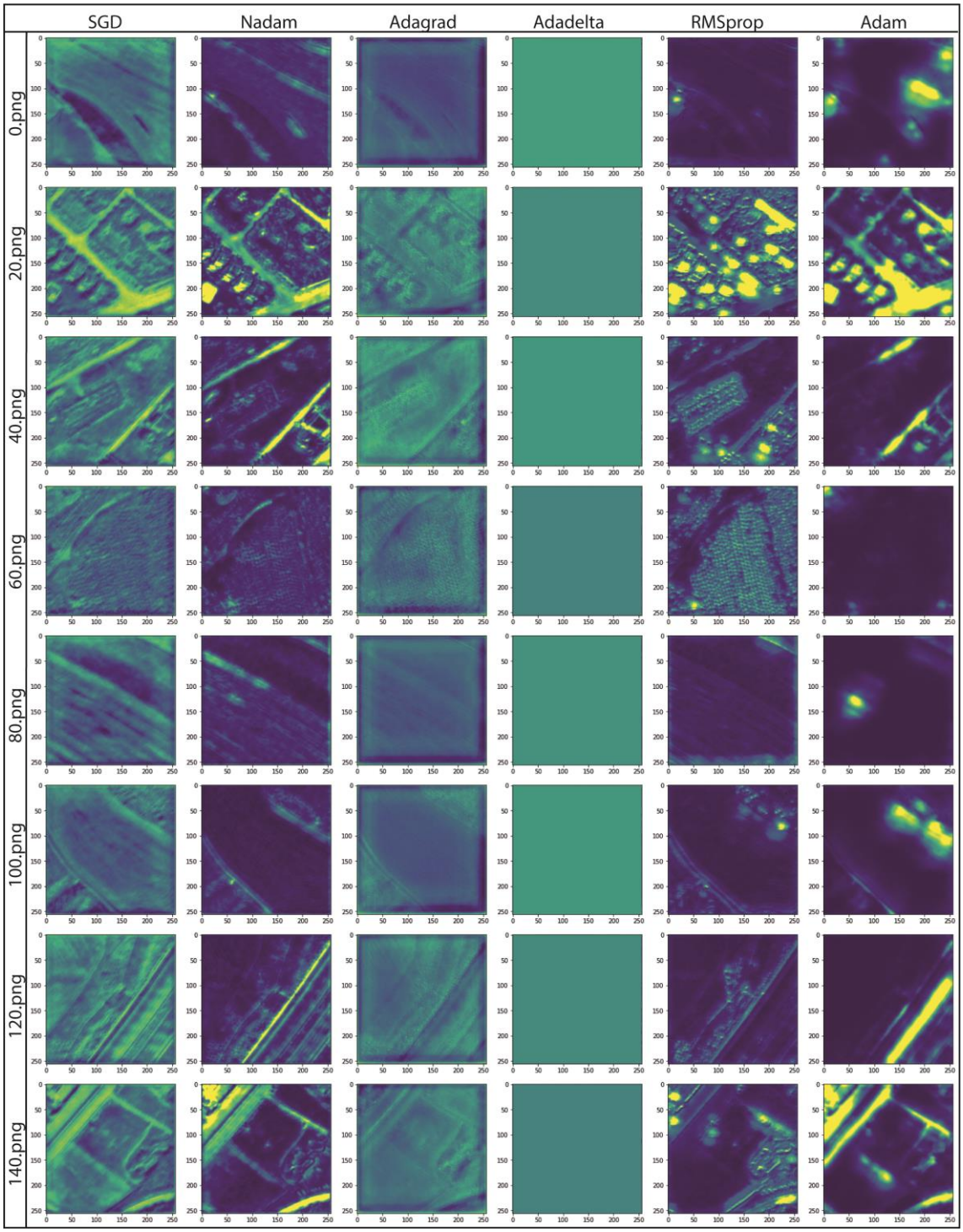
EK-8 Tiff-10 görüntülerinin 100 Epok Sonucunda tahmin edilen görüntüleri



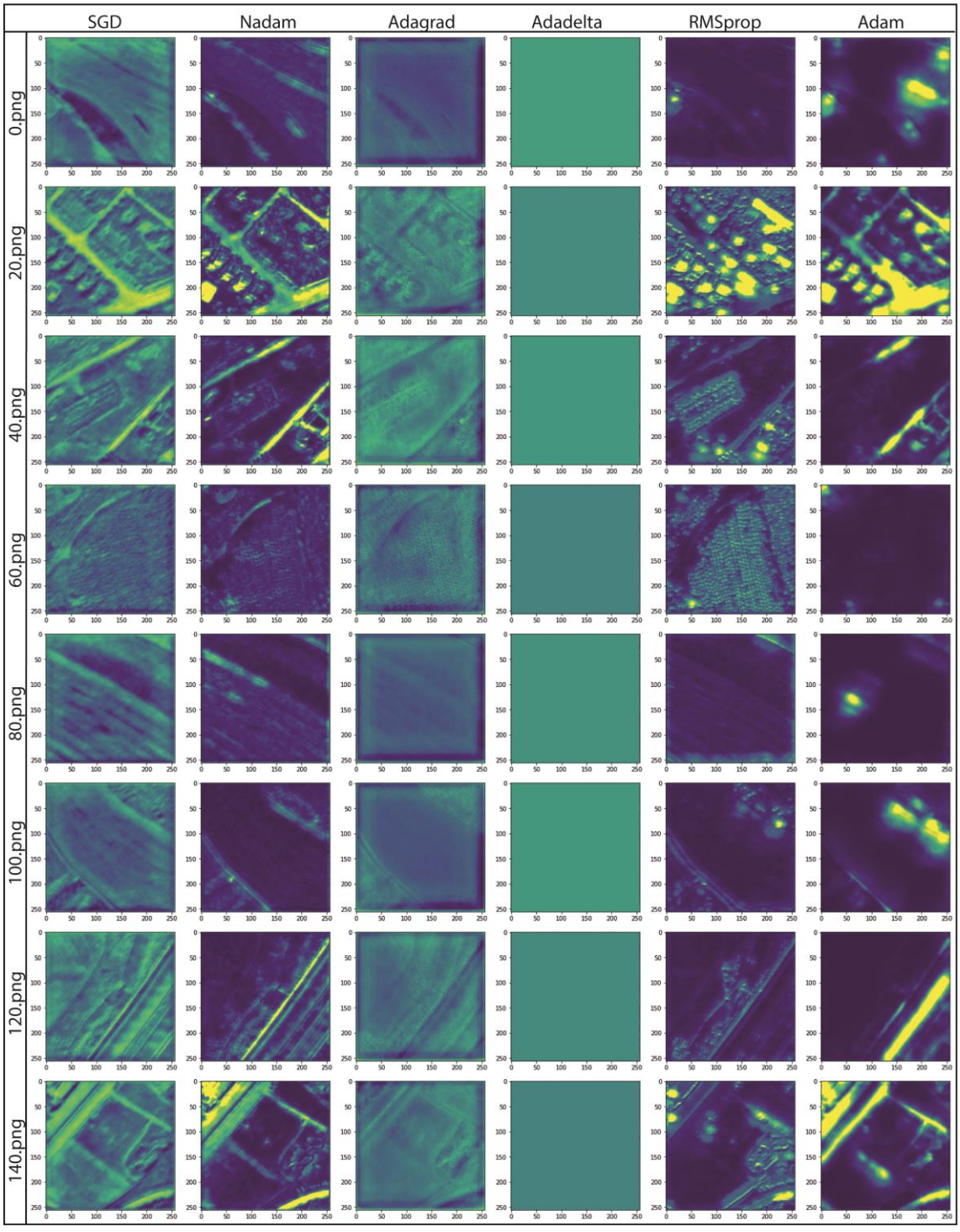
EK-9 Tiff-12 RGB ve Mask görüntüsü



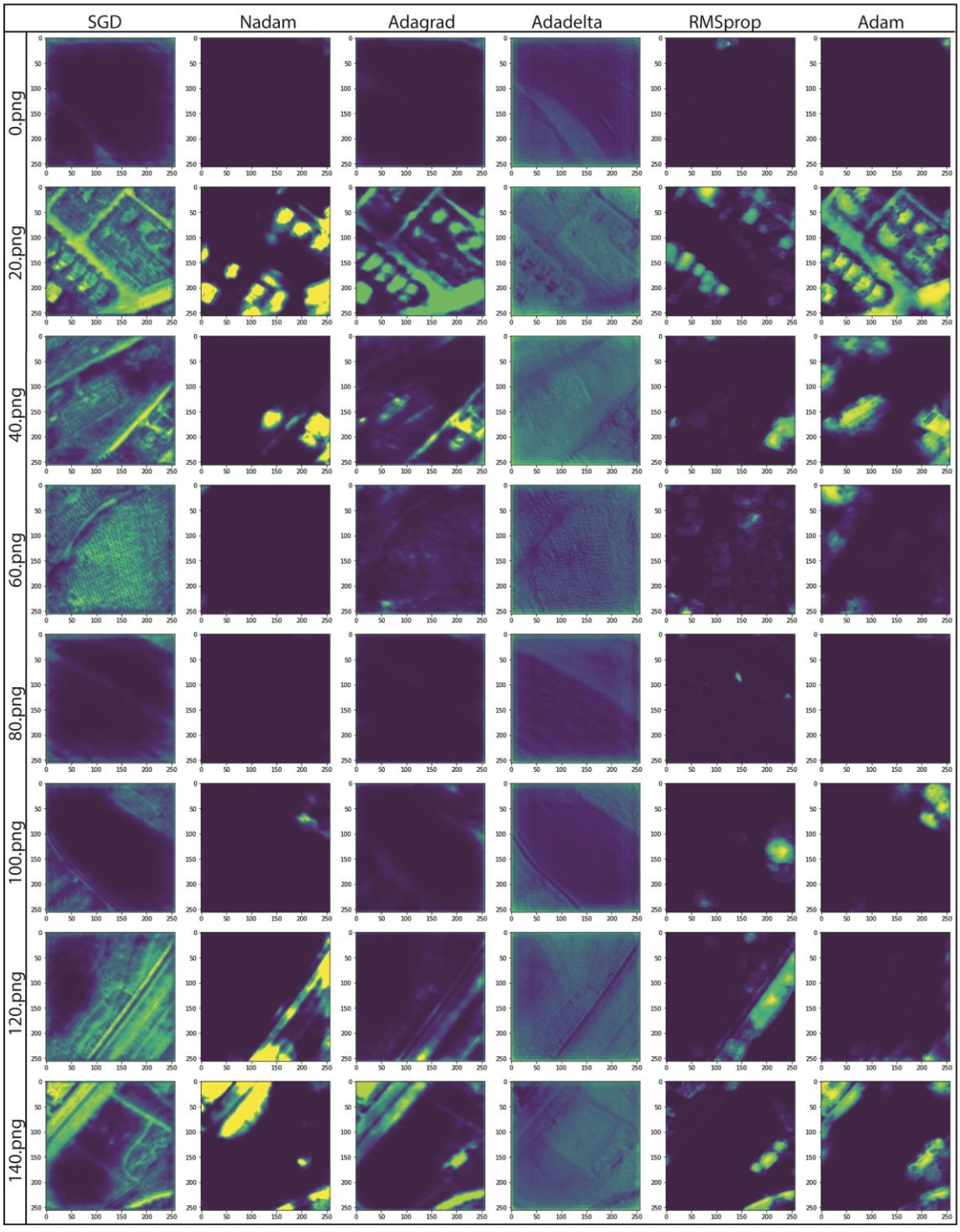
EK-10 Tiff-12 görüntülerinin 10 Epok Sonucunda tahmin edilen görüntüleri



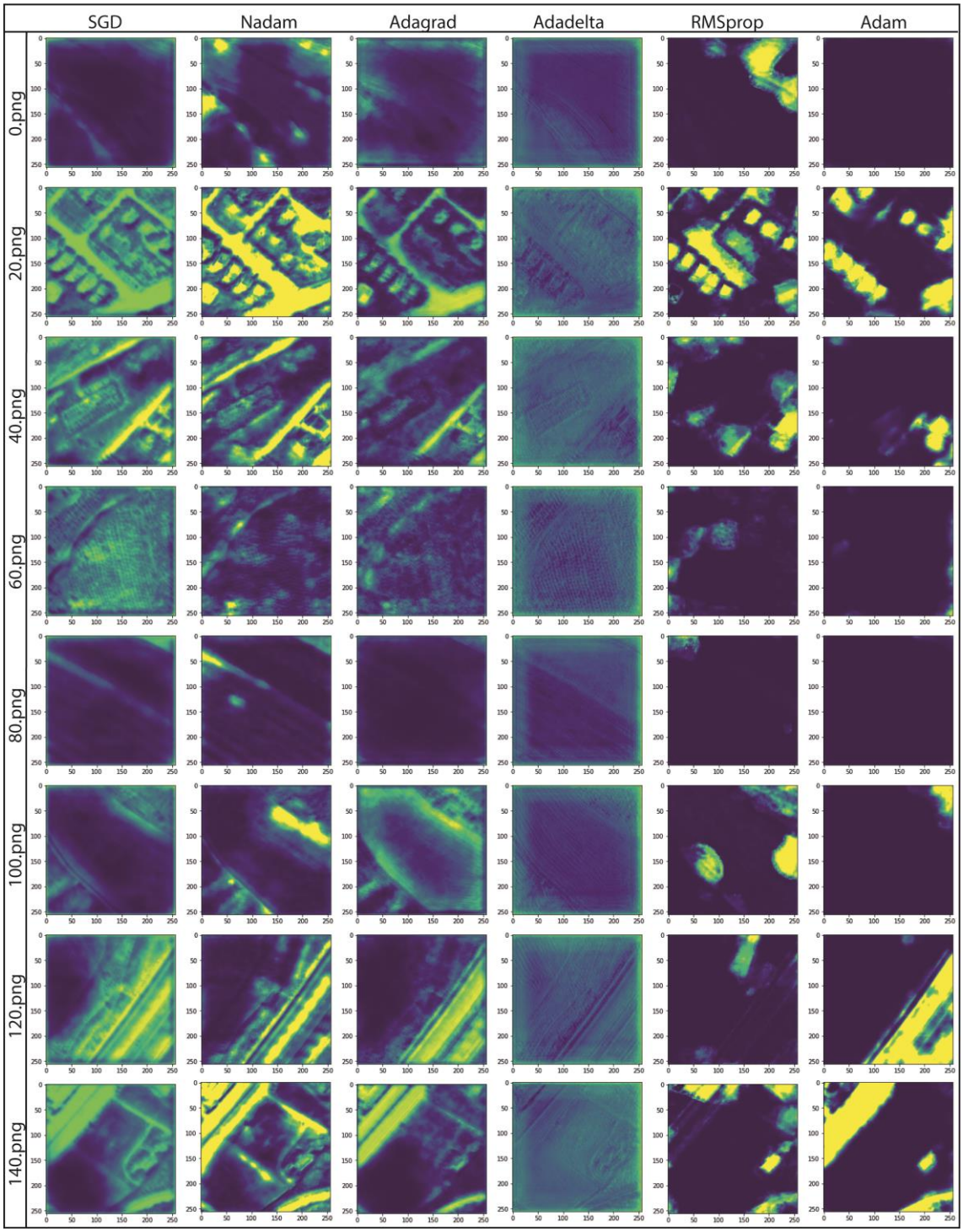
EK-11 Tiff-12 görüntülerinin 25 Epok Sonucunda tahmin edilen görüntüleri



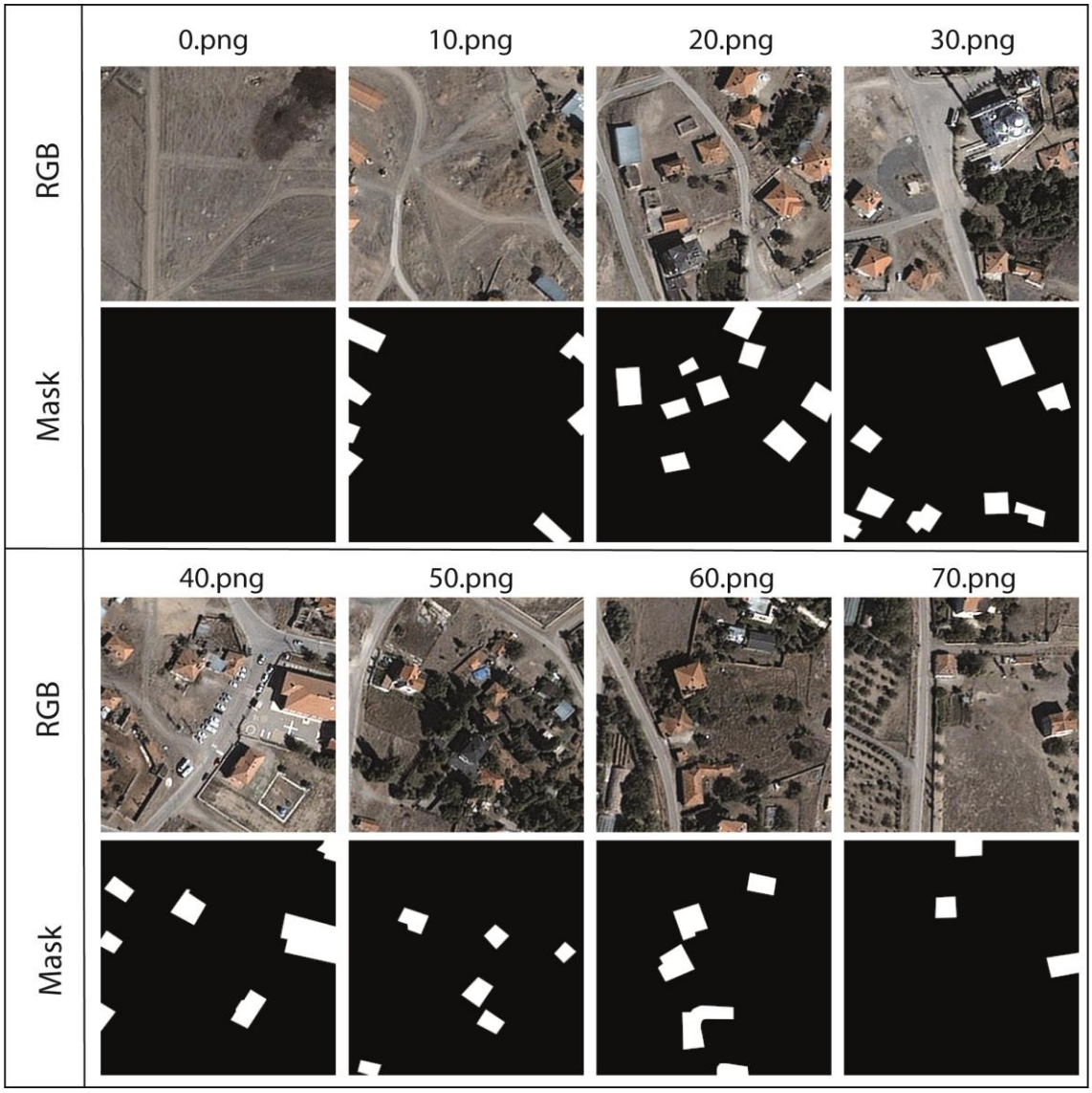
EK-12 Tiff-12 görüntülerinin 50 Epok Sonucunda tahmin edilen görüntüleri



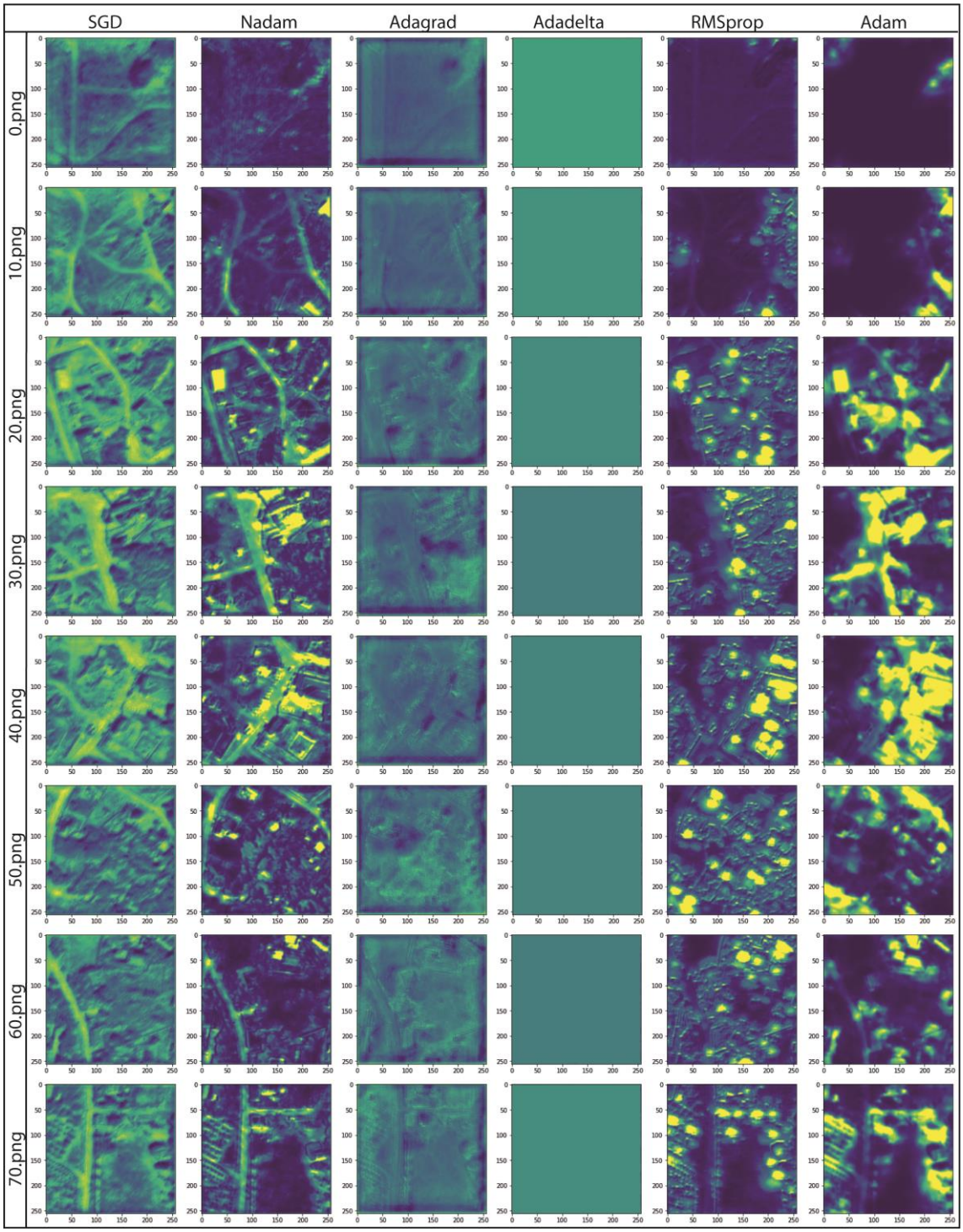
EK-13 Tiff-12 görüntülerinin 100 Epok Sonucunda tahmin edilen görüntüleri



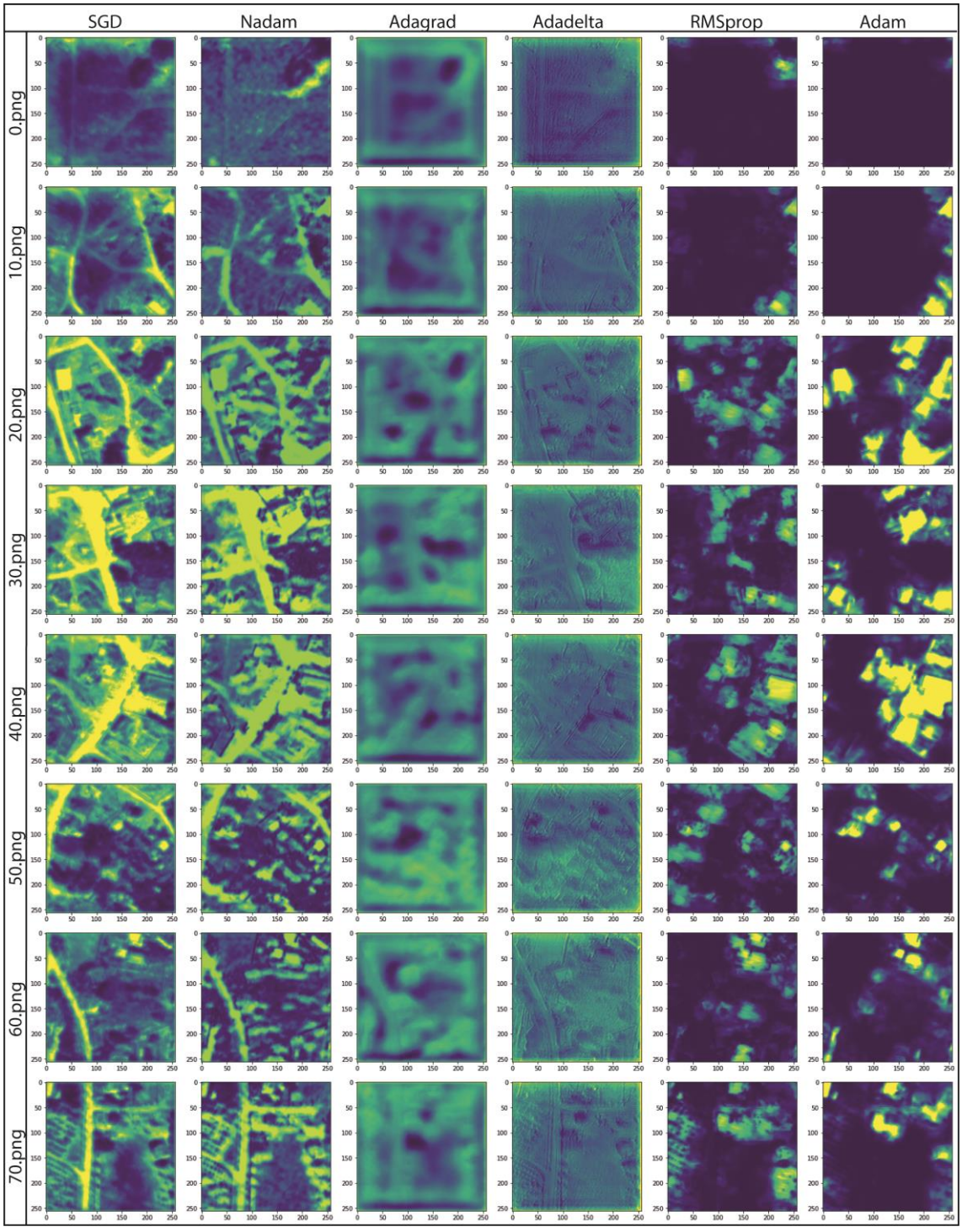
EK-14 Tiff-13 RGB ve Mask görüntüsü



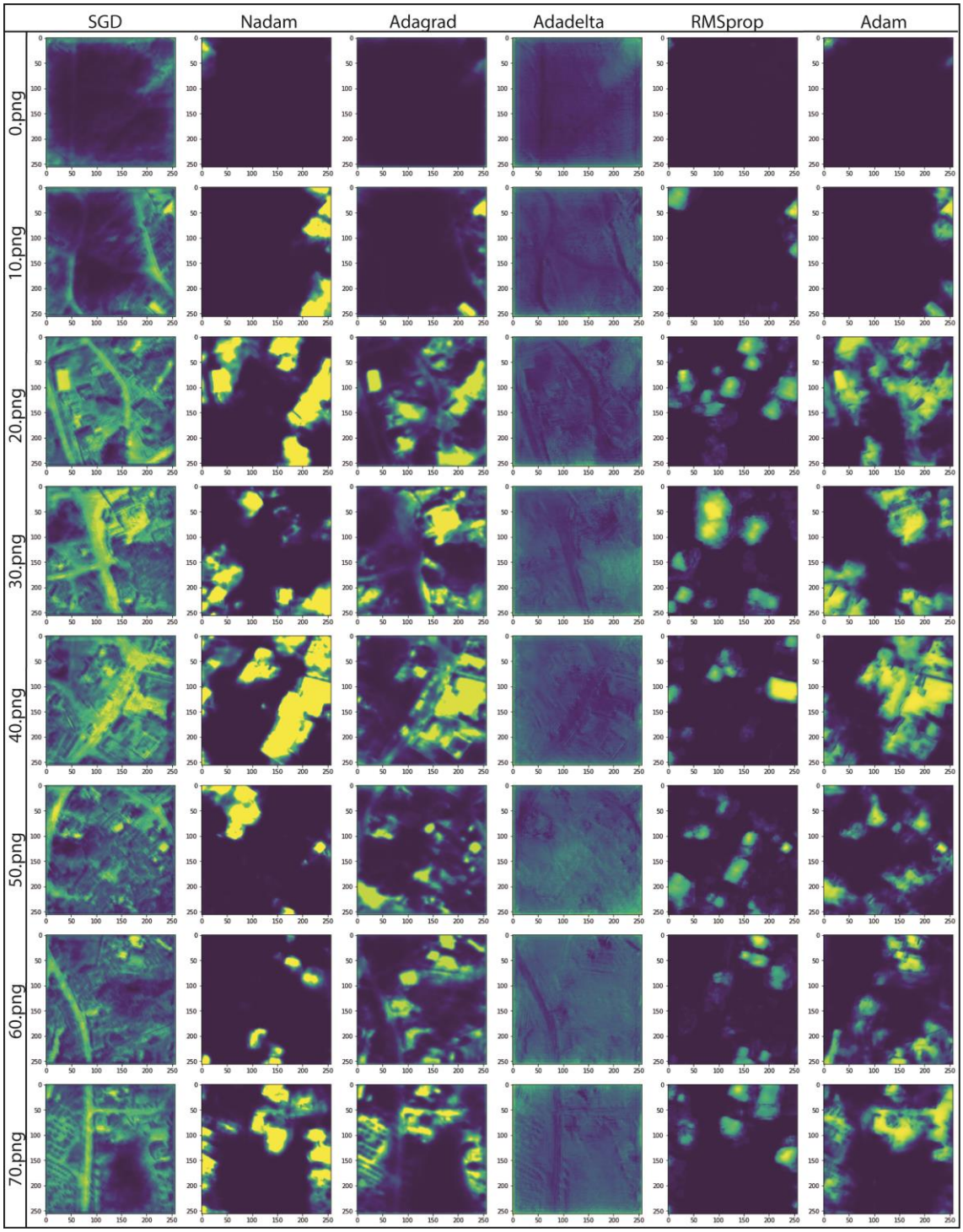
EK-15 Tiff-13 görüntülerinin 10 Epok Sonucunda tahmin edilen görüntüleri



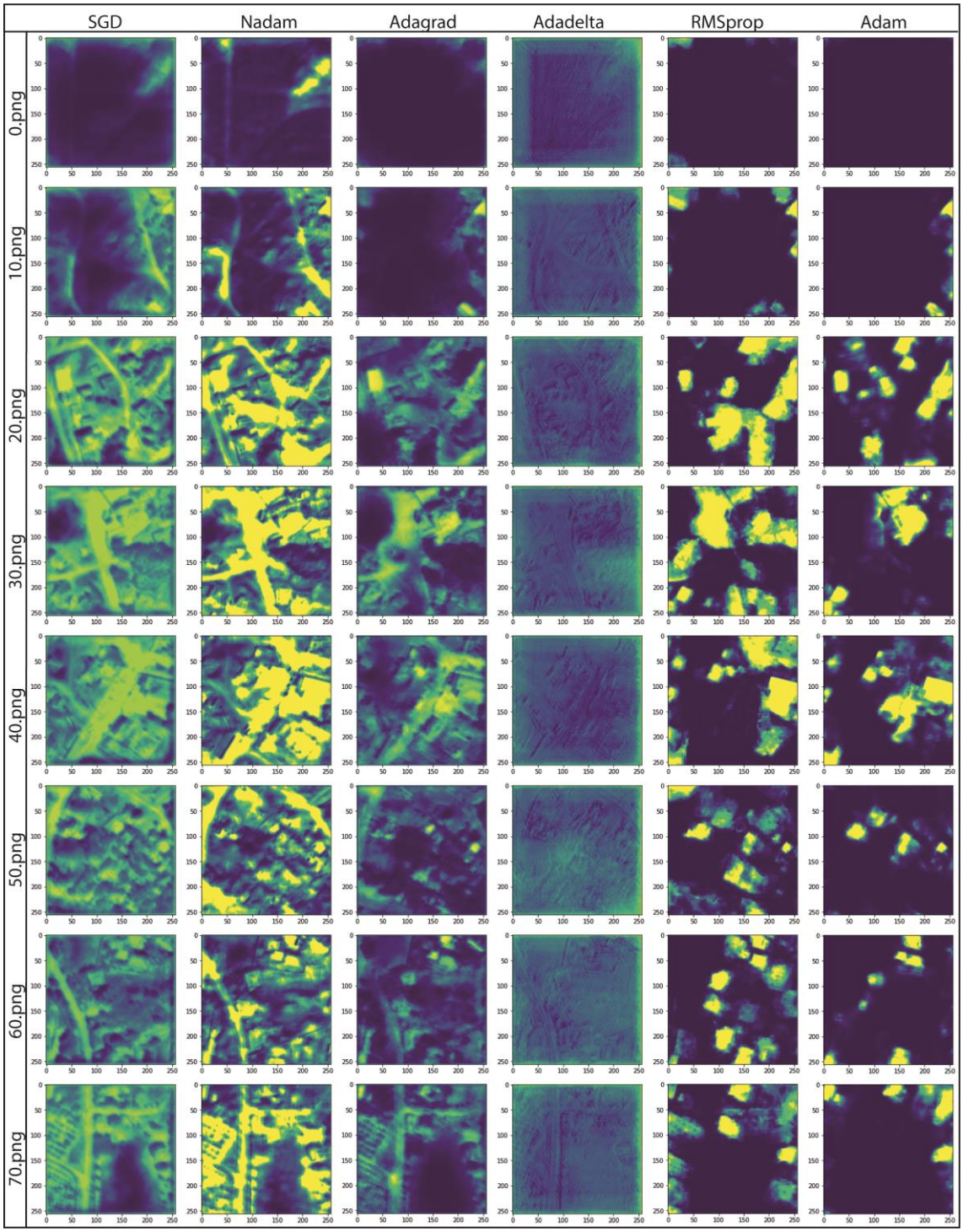
EK-16 Tiff-13 görüntülerinin 25 Epok Sonucunda tahmin edilen görüntüleri



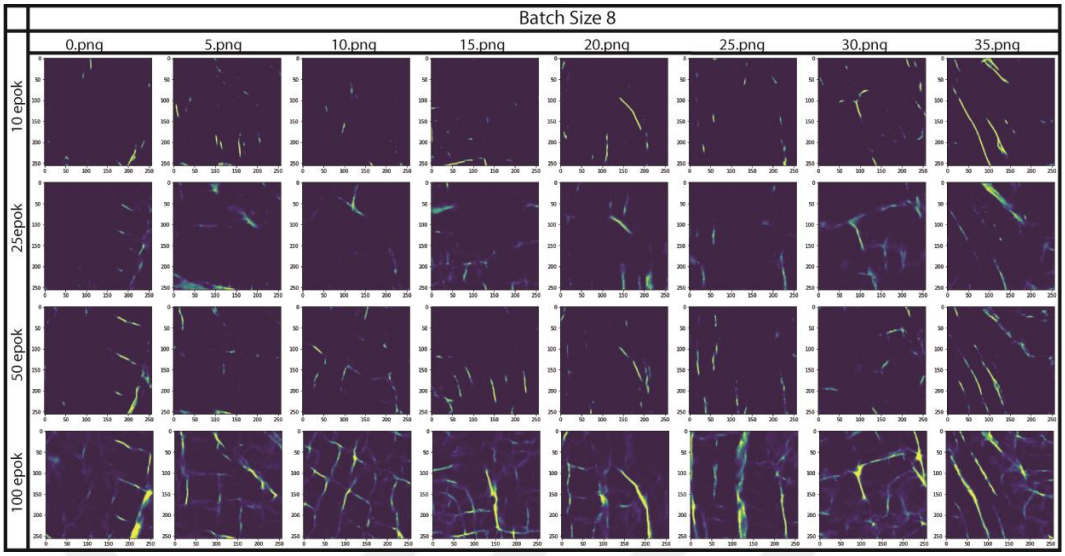
EK-17 Tiff-13 görüntülerinin 50 Epok Sonucunda tahmin edilen görüntüleri



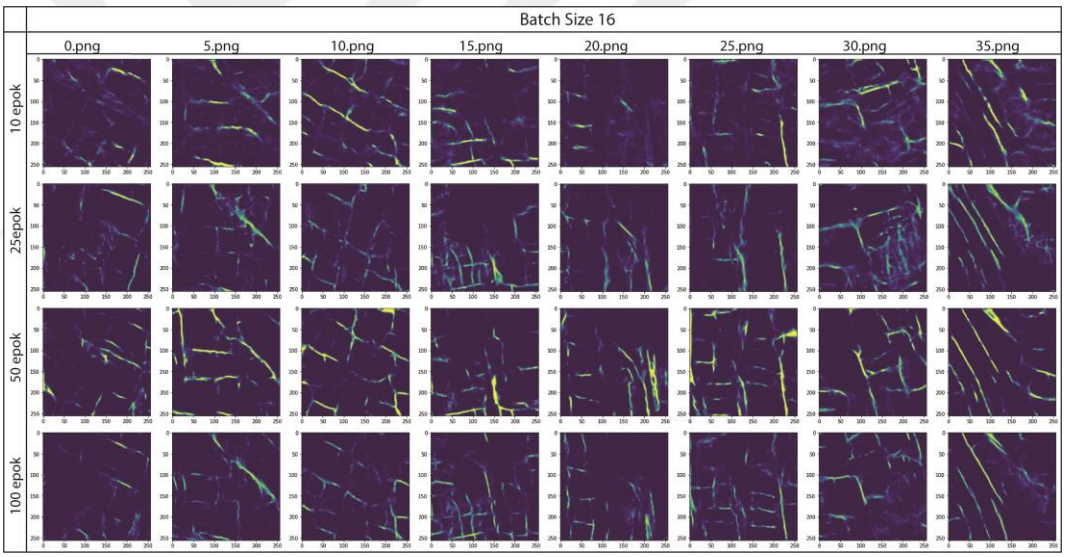
EK-18 Tiff-13 görüntülerinin 100 Epok Sonucunda tahmin edilen görüntüleri



EK-19 Tiff-10 görüntülerinin BS_8 Sonucunda Tahmin Edilen Görüntüleri



EK-20 Tiff-10 görüntülerinin BS_16 Sonucunda Tahmin Edilen Görüntüleri



EK-21 Tiff-10 görüntülerinin BS_32 Sonucunda Tahmin Edilen Görüntüleri