
**DERİN SİNİR AĞLARI VE UYGULAMALARI**

**FİNAL PROJE ÖDEVİ**

**U-NET MODELİ İLE RETİNA DAMARLARI SEGMENTASYONU**

Öğretim Görevlisi : Prof. Dr. Necaattin BARIŞÇI

Öğrenci No : 198330404

Öğrenci : Gamze GÜNAY

Tarih:16.06.2020


## Özet:

Göz doktorunun retinayı taraması, hipertansiyon, kardiyovasküler ve diyabet gibi potansiyel göz hastalıklarının erken belirtilerini saptamasını sağlar. Retina kamera görüntülerinin manuel olarak değerlendirilmesi zordur ve çeşitli hastalardan çok sayıda görüntü alındığında manuel olarak değerlendirmesi olanaksız hale gelir. Bu çalışmada bu sorunları çözmek için, geliştirilmiş derin öğrenme U-Net modeline dayanan bir fundus retina damarları segmentasyonu gerçekleştirilmesi amaçlanmıştır.Drive Data seti üzerinde U-Net modeli ile fundus retina damarları segmentasyonu ile gerçekleştirilmiştir. Deneysel sonuçlar ile DRIVE veri setinde retinal damar segmentasyonu için doğruluk, duyarlılık ve özgüllük değerleri sırasıyla % 97,32,% 99,73 ve% 97,85 olarak elde edilmiştir. Deneysel sonuçlar diğer yöntemler ile karşılaştırılmış ve diğer çalışmalara göre ROC curve (AUC) altındaki alan değerlendirildiğinde 0.9772 AUC ile DRIVE datasetinde başarılı performans elde ettiği gözlemlenmiştir.


## Giriş:

Diyabet ve hipertansiyon gibi bu hastalıklar retinanın kan damarlarında morfolojik değişikliklere neden olur bu nedenle Birçok hastalık fundus vasküler sistemi gözlenerek teşhis edilebilir ve izlenebilir. Retina damar segmentasyonu fundus görüntülerinin kantitatif analizinde önemli bir adımdır. Retina kan damarlarını bölümlere ayırarak, retina kan damarı ağacının (kan damarlarının eğriliği, uzunluğu ve genişliği gibi) ilgili morfolojik bilgilerini elde edebiliriz [1]. Ayrıca, retinal damarların vasküler ağacı benzersiz özelliklere sahiptir ve biyometrik tanıma için de uygulanabilir [2] [3]. Bu nedenle, retina kan damarlarının doğru segmentasyonu büyük önem taşımaktadır. Tıbbi görüntü segmentasyonu alanında, U-Net [4] yaygın ve iyi bilinen bir derin öğrenme ağıdır.

Temel olarak, U-Net tipik bir altörnekleme kodlayıcısı ve üst örnekleme kod çözücü yapısı ile aralarındaki bağlantıdan oluşur. Bu bağlantı yerel ve global bilgileri kodlama ve kod çözme işlemiyle birleştirir. Wang ve diğ. ağın retina damarlarını uçtan uca ve pikselden piksele bölümlere ayırma yeteneğini önemli ölçüde artıran Çift Kodlamalı U-Net'i (DEU-Net) bildirmiştir [5]. Wu ve diğ. ilk kez retinal damar segmentasyonu gerçekleştirmek için Vessel-Net'i önermiştir [6]. Y. Guo ve diğ. Multiple Deep Convolutional Neural Network (MDCNN) ile retina kan damarları segmentasyonunu gerçekleşmiştir. Önerilen MDCNN, geleneksel CNN'ye göre performanslarını artırmak için artımlı bir strateji kullanılarak eğitilmiştir [7]. P. Xiuqin ve diğ. U-Net modeline dayanan bir fundus retina damarları segmentasyonu geliştirilmiştir[8].U-Net modeline, ResNET mimarisinden gelen residual bağlantılar eklenerek ağın derinliği arttırılmıştır. Z. Yan ve diğ. üç aşamalı derin öğrenme mimarisi önerek kalın damar segmentasyonu, ince damar segmentasyonu ve damar füzyonu olmak üzere üç aşama ile segmentasyon gerçekleştirilmiştir [9].

Bu çalışmada, geliştirilmiş derin öğrenme U-Net modeline dayanan bir fundus retina damarları segmentasyonu gerçekleştirilmesi amaçlanmıştır.Drive Data seti üzerinde U-Net modeli ile fundus retina damarları segmentasyonu ile gerçekleştirilmiştir. Deneysel sonuçlar ile DRIVE veri setinde retinal damar segmentasyonu için doğruluk, duyarlılık ve özgüllük değerleri sırasıyla % 97,32,% 99,73 ve% 97,85 olarak elde edilmiştir. Deneysel sonuçlar diğer yöntemler ile karşılaştırılmış ve diğer çalışmalara göre ROC curve (AUC) altındaki alan değerlendirildiğinde 0.9772 AUC ile DRIVE datasetinde başarılı performans elde ettiği gözlemlenmiştir.

## Yöntem:

**Drive Veri Tabanı**

Digital Retinal Images for Vessel Extraction (DRIVE) veritabanı, 25-90 yaş aralığında 400 civarında diyabetik retinopati vakasından rastgele seçilen 40 adet renkli retinal fundus görüntüden oluşmaktadır [10]. Bu görüntüler 45 derece görüş açısında ve 768x584 piksel çözünürlükte ve 24 bit renk derinliğine sahiptir. 40 görüntüden 33 adedi herhangi bir patoloji içermezken 7 tanesi erken dönem diyabetik retinopati vakası içermektedir.

<u>Eğitim Görüntüleri erken dönem diyabetik retinopati vakası içeren görüntüler:</u>

25_training: pigment epithelium changes, probably butterfly maculopathy with pigmented scar in fovea, or choroidiopathy, no diabetic retinopathy or other vascular abnormalities.
26_training: background diabetic retinopathy,
32_training: background diabetic retinopathy

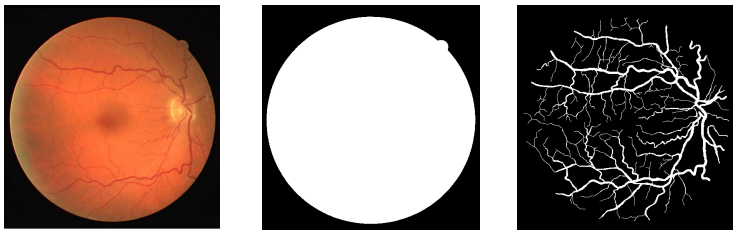
Test Görüntüleri erken dönem diyabetik retinopati vakası içeren görüntüler

03_test: background diabetic retinopathy
08_test: pigment epithelium changes, pigmented scar in fovea, or choroidiopathy, no diabetic retinopathy or other vascular abnormalities
14_test: background diabetic retinopathy
17_test: background diabetic retinopathy

Veritabanındaki görüntüler 2 alt kümeye ayrılmıştır.

- İlk 20 görüntü eğitim görüntüleri olarak kullanılmıştır.
- İkinci 20 görüntü ise test görüntüsü olarak kullanılmıştır.

Bu veritabanı için, görüntüler retina görüntüsü etrafında kırpılmıştır. Her görüntü için, retina görüntüsü tanımlayan bir maske görüntüsü sağlanır. Eğitim görüntüleri ve test görüntüleri için manuel segmentasyonu (Ground Truth) verilmiştir.



**Şekil 1** Orjinal görüntü/Maske Görüntüsü/Ground Truth

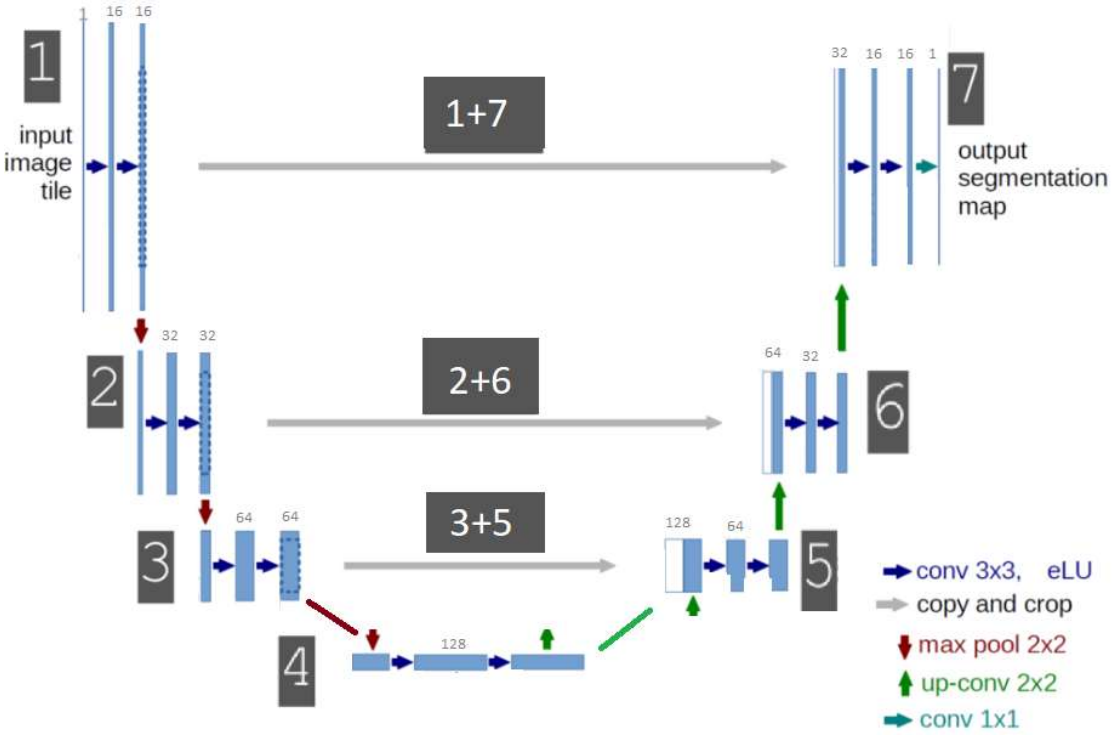
**Pre-Processing**

Derin öğrenme U-Net modelini öncesinde drive database verileri ön işleme tabi tutulmuştur. Sırasıyla:

1. Orjinal DRIVE database den alınmış görüntü
2. RGB to gray
3. Normalization
4. Histogram Equalization (CLAHE) görüntüde histogram eşitleme gerçekleştirilmiştir.
5. Gamma correction işlemi yapılmıştır.

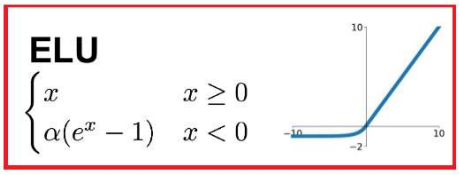
**Network Yapisi**

Bu çalışmada, Şekil 2'de gösterildiği gibi damar segmentasyonu için U-Net mimarisini kullanılmıştır. U-Net mimarisi kodlama, köprü ve kod çözme olmak üzere üç bölümden oluşmaktadır.




**Şekil 2** Retina damar segmentasyonu için çalışmada kullanılan U-Net mimarisi

Şekil 2, bir U-NET mimarisinin kodlayıcı bölümüne her adımı 2x2 max pooling işlemi uygulanmıştır . Her bir convolution katmanı arasında bir DropBlock blok 0.1 olarak uygulanmıştır. Her bir convolution işleminden sonra eLU aktivasyon fonksiyonu uygulanmıştır.




$$f(\alpha, x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$$

**Şekil 3** eLU aktivasyon fonksiyonu

U-NET mimarisinin kodlayıcı bölümüne convolution işlemleri sırası ile 16, 32,64 adet 3x3 filtre ile uygulanmıştır. U-NET mimarisinin köprü bölümüne her bir convolution bloğun evrişimsel katmanı, eLU aktivasyon konksiyonundan geçirilir ve iki convolution blok arasında Dropout blok 0.1 olarak uygulanmıştır. Kod çözücü bölümünde her adımda upsampling işlemi ile feature kanal sayısı iki katına çıkarılır. Kodlayıcı bölümünden karşılık gelen feature map ile birleştirilerek, convolution katmanından geçirilir. Her bir convolution katmanı arasında bir DropBlock blok 0.1 ve 0.2 olarak uygulanmıştır. U-NET mimarisinin kod çözücü bölümüne convolution işlemleri sırası ile 64, 32,16 adet 3x3 filtre ile uygulanmıştır. Son katmanda 1 × 1 convolution ve Sigmoid aktivasyon fonksiyonu ile segmentasyon gerçekleştirilmiştir.

```
# 1\ Down 1
conv1 = Conv2D(16, (3, 3), activation='elu', padding='same')(inputs)
conv1 = Dropout(0.1)(conv1)
conv1 = Conv2D(16, (3, 3), activation='elu', padding='same')(conv1)
pool1 = MaxPooling2D((2, 2))(conv1)

# 2\ Down 2
conv2 = Conv2D(32, (3, 3), activation='elu', padding='same')(pool1)
conv2 = Dropout(0.1)(conv2)
conv2 = Conv2D(32, (3, 3), activation='elu', padding='same')(conv2)
pool2 = MaxPooling2D((2, 2))(conv2)

# 3\ Down 3
conv3 = Conv2D(64, (3, 3), activation='elu', padding='same')(pool2)
conv3 = Dropout(0.1)(conv3)
conv3 = Conv2D(64, (3, 3), activation='elu', padding='same')(conv3)
pool3 = MaxPooling2D((2, 2))(conv3)

# 3\ Middle
conv4 = Conv2D(128, (3, 3), activation='elu', padding='same')(pool3)
conv4 = Dropout(0.1)(conv4)
conv4 = Conv2D(128, (3, 3), activation='elu', padding='same')(conv4)

# 4\ Up 1
up1 = UpSampling2D(size=(2, 2))(conv4)
up1 = Concatenate(axis=-1)([conv3,up1])
conv5 = Conv2D(64, (3, 3), activation='elu', padding='same')(up1)
conv5 = Dropout(0.1)(conv5)
conv5 = Conv2D(64, (3, 3), activation='elu', padding='same')(conv5)

# 6\ Up 2
up2 = UpSampling2D(size=(2, 2))(conv5)
up2 = Concatenate(axis=-1)([conv2,up2])
conv6 = Conv2D(32, (3, 3), activation='elu', padding='same')(up2)
conv6 = Dropout(0.2)(conv6)
conv6 = Conv2D(32, (3, 3), activation='elu', padding='same')(conv6)

# 7\ Up 3
up3 = UpSampling2D(size=(2, 2))(conv6)
up3 = Concatenate(axis=-1)([conv1, up3])
conv7 = Conv2D(16, (3, 3), activation='elu', padding='same')(up3)
conv7 = Dropout(0.2)(conv7)
conv7 = Conv2D(16, (3, 3), activation='elu', padding='same')(conv7)

# 8\ Final
conv9 = Conv2D(1, (1, 1), activation='sigmoid', padding='same')(conv7)
outputs = conv9
```

**Şekil 4** Retina damar segmentasyonu için  çalışmada kullanılan U-Net mimarisi

**Ağirliklarin Güncellenmesi :**

Loss fonksiyonunun optimize edilip ağırlıkların ayarlanmasında  Adaptive Moment Optimization (Adam) optimizasyon algoritması kullanılmıştır. Learning_rate=0.001 başlangıçta seçilmiştir 5 epoch ta bir düşme faktörü 0,005  düşürülmüştür. beta_1: 0.9, beta_2: 0.999, epsilon1e-7 olarak alınmıştır.

$$\nu_t = \beta_1 * \nu_{t-1} - (1 - \beta_1) * g_t$$
$$s_t = \beta_2 * s_{t-1} - (1 - \beta_2) * g_t^2$$
$$\Delta\omega_t = -\eta \frac{\nu_t}{\sqrt{s_t + \epsilon}} * g_t$$
$$\omega_{t+1} = \omega_t + \Delta\omega_t$$

$\eta : Initial\ Learning\ rate$
$g_t : Gradient\ at\ time\ t\ along\ \omega^j$
$\nu_t : Exponential\ Average\ of\ gradients\ along\ \omega_j$
$s_t : Exponential\ Average\ of\ squares\ of\ gradients\ along\ \omega_j$
$\beta_1, \beta_2 : Hyperparameters$

**Şekil 5** Adaptive Moment Optimization (Adam) optimizasyonu

**Loss Fonksiyonu**

Kayıp fonksiyonunun hesaplanmasında ikili çapraz entropi (binary cross entropy) kullanılmıştır. 200000 adet patch (48x48x1) ile 20 epoch boyunca eğitim gerçekleştirilmiştir. Batch size 128 olarak uygulanmıştır.

$$L_{bce} = \sum_i y_i \log o_i + (1 - y_i) \log(1 - o_i)$$

**Doğruluk, Duyarlılık Ve Özgüllük VE F1 Score Değerleri Hesaplanması**

Sonıflandırma sonucunda U-NET modelinin tahminleri test verisi ile kıyaslandığında, aşağidaki gibi değerlendirilir:

- U-NET modeli 1 olarak tahmin etmiş ve test verisi 1 ise : Doğru Pozitif (True Positive)(TP)
- U-NET modeli 0 olarak tahmin etmiş ve test verisi 0 ise : Doğru Negatif (True Negative)(TN)
- U-NET modeli 1 olarak tahmin etmiş ve test verisi 0 ise : Yanlış Pozitif (False Positive)(FP)
- U-NET modeli 0 olarak tahmin etmiş ve test verisi 1 ise : Yanlış Negatif (False Negative)(FN)

  - $Doğruluk = TP + TN\ (TP + FP + FN + TN)$
  - $Duyarlılık = TP/(TP + FN)$
  - Özgüllük= $TN/(TN + FP)$
  - $F1Score = 2TP/(2TP + FP + FN)$

**20. Epoch Sonunda Doğruluk, Duyarlılık Ve Özgüllük VE F1 Score Değerleri**

DRIVE data seti ile U-Net modelinin 20. epoch sonunda doğruluk, duyarlılık ve özgüllük değerleri sırasıyla% 97,32,% 99,73 ve % 97,85 elde edilmiştir.

GPU count:1, Memory growth:True, Soft device placement:True ...
Train on 180000 samples, validate on 20000 samples
Epoch 1/20
Epoch 00001: val_loss improved from inf to 0.11718, saving model to ./result/config_0/best_weights.h5
180000/180000 [==============================] - 454s 3ms/sample - loss: 0.2025 - accuracy: 0.9293 - f1_m: 0.4942 - recall_m: 0.4159 - precision_m: 0.7008 - sensitivity_at_specificity: 0.9586 - specificity: 0.9354 - val_loss: 0.1172 - val_accuracy: 0.9550 - val_f1_m: 0.7419 - val_recall_m: 0.6841 - val_precision_m: 0.8129 - val_sensitivity_at_specificity: 0.9951 - val_specificity: 0.9663
Epoch 2/20
Epoch 00002: val_loss improved from 0.11718 to 0.10045, saving model to ./result/config_0/best_weights.h5
180000/180000 [==============================] - 448s 2ms/sample - loss: 0.1279 - accuracy: 0.9526 - f1_m: 0.7437 - recall_m: 0.6758 - precision_m: 0.8289 - sensitivity_at_specificity: 0.9939 - specificity: 0.9598 - val_loss: 0.1004 - val_accuracy: 0.9611 - val_f1_m: 0.7932 - val_recall_m: 0.7885 - val_precision_m: 0.7992 - val_sensitivity_at_specificity: 0.9978 - val_specificity: 0.9587
Epoch 3/20
Epoch 00003: val_loss improved from 0.10045 to 0.09602, saving model to ./result/config_0/best_weights.h5
180000/180000 [==============================] - 448s 2ms/sample - loss: 0.1094 - accuracy: 0.9588 - f1_m: 0.7828 - recall_m: 0.7288 - precision_m: 0.8465 - sensitivity_at_specificity: 0.9929 - specificity: 0.9655 - val_loss: 0.0960 - val_accuracy: 0.9615 - val_f1_m: 0.7993 - val_recall_m: 0.8099 - val_precision_m: 0.7894 - val_sensitivity_at_specificity: 0.9948 - val_specificity: 0.9641
Epoch 4/20
Epoch 00004: val_loss did not improve from 0.09602
180000/180000 [==============================] - 447s 2ms/sample - loss: 0.0972 - accuracy: 0.9626 - f1_m: 0.8058 - recall_m: 0.7618 - precision_m: 0.8560 - sensitivity_at_specificity: 0.9932 - specificity: 0.9691 - val_loss: 0.0982 - val_accuracy: 0.9611 - val_f1_m: 0.8038 - val_recall_m: 0.8413 - val_precision_m: 0.7698 - val_sensitivity_at_specificity: 0.9955 - val_specificity: 0.9593
Epoch 5/20
Epoch 00005: val_loss improved from 0.09602 to 0.09229, saving model to ./result/config_0/best_weights.h5

180000/180000 [==============================] - 448s 2ms/sample - loss: 0.1259 - accuracy: 0.9532 - f1_m: 0.7346 - recall_m: 0.6812 - precision_m: 0.8233 - sensitivity_at_specificity: 0.9938 - specificity: 0.9606 - val_loss: 0.0923 - val_accuracy: 0.9629 - val_f1_m: 0.8055 - val_recall_m: 0.8114 - val_precision_m: 0.8000 - val_sensitivity_at_specificity: 0.9973 - val_specificity: 0.9628
Epoch 6/20
Epoch 00006: val_loss did not improve from 0.09229
180000/180000 [==============================] - 446s 2ms/sample - loss: 0.0966 - accuracy: 0.9627 - f1_m: 0.8065 - recall_m: 0.7637 - precision_m: 0.8550 - sensitivity_at_specificity: 0.9932 - specificity: 0.9692 - val_loss: 0.0947 - val_accuracy: 0.9618 - val_f1_m: 0.8068 - val_recall_m: 0.8415 - val_precision_m: 0.7753 - val_sensitivity_at_specificity: 0.9956 - val_specificity: 0.9604
Epoch 7/20
Epoch 00007: val_loss did not improve from 0.09229
180000/180000 [==============================] - 446s 2ms/sample - loss: 0.0895 - accuracy: 0.9648 - f1_m: 0.8191 - recall_m: 0.7821 - precision_m: 0.8601 - sensitivity_at_specificity: 0.9940 - specificity: 0.9712 - val_loss: 0.0967 - val_accuracy: 0.9613 - val_f1_m: 0.8049 - val_recall_m: 0.8426 - val_precision_m: 0.7707 - val_sensitivity_at_specificity: 0.9930 - val_specificity: 0.9614
Epoch 8/20
Epoch 00008: val_loss did not improve from 0.09229
180000/180000 [==============================] - 446s 2ms/sample - loss: 0.0852 - accuracy: 0.9661 - f1_m: 0.8263 - recall_m: 0.7923 - precision_m: 0.8637 - sensitivity_at_specificity: 0.9947 - specificity: 0.9723 - val_loss: 0.0954 - val_accuracy: 0.9616 - val_f1_m: 0.8048 - val_recall_m: 0.8344 - val_precision_m: 0.7777 - val_sensitivity_at_specificity: 0.9929 - val_specificity: 0.9625
Epoch 9/20
Epoch 00009: val_loss did not improve from 0.09229
180000/180000 [==============================] - 446s 2ms/sample - loss: 0.0808 - accuracy: 0.9674 - f1_m: 0.8337 - recall_m: 0.8025 - precision_m: 0.8678 - sensitivity_at_specificity: 0.9953 - specificity: 0.9736 - val_loss: 0.0974 - val_accuracy: 0.9618 - val_f1_m: 0.8061 - val_recall_m: 0.8375 - val_precision_m: 0.7774 - val_sensitivity_at_specificity: 0.9904 - val_specificity: 0.9639
Epoch 10/20
Epoch 00010: val_loss did not improve from 0.09229
180000/180000 [==============================] - 446s 2ms/sample - loss: 0.0767 - accuracy: 0.9686 - f1_m: 0.8406 - recall_m: 0.8119 - precision_m: 0.8717 - sensitivity_at_specificity: 0.9959 - specificity: 0.9747 - val_loss: 0.1024 - val_accuracy: 0.9604 - val_f1_m: 0.8012 - val_recall_m: 0.8424 - val_precision_m: 0.7641 - val_sensitivity_at_specificity: 0.9903 - val_specificity: 0.9619
Epoch 11/20
Epoch 00011: val_loss did not improve from 0.09229
180000/180000 [==============================] - 446s 2ms/sample - loss: 0.0810 - accuracy: 0.9674 - f1_m: 0.8334 - recall_m: 0.8018 - precision_m: 0.8682 - sensitivity_at_specificity: 0.9953 - specificity: 0.9735 - val_loss: 0.1024 - val_accuracy: 0.9603 - val_f1_m: 0.8008 - val_recall_m: 0.8426 - val_precision_m: 0.7634 - val_sensitivity_at_specificity: 0.9894 - val_specificity: 0.9630
Epoch 12/20
Epoch 00012: val_loss did not improve from 0.09229
180000/180000 [==============================] - 448s 2ms/sample - loss: 0.0767 - accuracy: 0.9687 - f1_m: 0.8408 - recall_m: 0.8121 - precision_m: 0.8721 - sensitivity_at_specificity: 0.9959 - specificity: 0.9747 - val_loss: 0.0983 - val_accuracy: 0.9614 - val_f1_m: 0.8048 - val_recall_m: 0.8391 - val_precision_m: 0.7737 - val_sensitivity_at_specificity: 0.9906 - val_specificity: 0.9644
Epoch 13/20
Epoch 00013: val_loss did not improve from 0.09229
180000/180000 [==============================] - 448s 2ms/sample - loss: 0.0724 - accuracy: 0.9701 - f1_m: 0.8485 - recall_m: 0.8225 - precision_m: 0.8764 - sensitivity_at_specificity: 0.9965 - specificity: 0.9759 - val_loss: 0.1020 - val_accuracy: 0.9605 - val_f1_m: 0.8014 - val_recall_m: 0.8412 - val_precision_m: 0.7654 - val_sensitivity_at_specificity: 0.9906 - val_specificity: 0.9619
Epoch 14/20
Epoch 00014: val_loss did not improve from 0.09229

180000/180000 [==============================] - 445s 2ms/sample - loss: 0.0698 - accuracy: 0.9710 - f1_m: 0.8535 - recall_m: 0.8294 - precision_m: 0.8792 - sensitivity_at_specificity: 0.9967 - specificity: 0.9767 - val_loss: 0.1066 - val_accuracy: 0.9599 - val_f1_m: 0.7994 - val_recall_m: 0.8429 - val_precision_m: 0.7605 - val_sensitivity_at_specificity: 0.9859 - val_specificity: 0.9641
Epoch 15/20
Epoch 00015: val_loss did not improve from 0.09229
180000/180000 [==============================] - 450s 2ms/sample - loss: 0.0697 - accuracy: 0.9711 - f1_m: 0.8542 - recall_m: 0.8301 - precision_m: 0.8799 - sensitivity_at_specificity: 0.9967 - specificity: 0.9768 - val_loss: 0.1029 - val_accuracy: 0.9606 - val_f1_m: 0.8010 - val_recall_m: 0.8371 - val_precision_m: 0.7682 - val_sensitivity_at_specificity: 0.9881 - val_specificity: 0.9641
Epoch 16/20
Epoch 00016: val_loss did not improve from 0.09229
180000/180000 [==============================] - 451s 3ms/sample - loss: 0.0673 - accuracy: 0.9719 - f1_m: 0.8587 - recall_m: 0.8363 - precision_m: 0.8824 - sensitivity_at_specificity: 0.9970 - specificity: 0.9775 - val_loss: 0.1051 - val_accuracy: 0.9599 - val_f1_m: 0.7996 - val_recall_m: 0.8459 - val_precision_m: 0.7585 - val_sensitivity_at_specificity: 0.9878 - val_specificity: 0.9626
Epoch 17/20
Epoch 00017: val_loss did not improve from 0.09229
180000/180000 [==============================] - 448s 2ms/sample - loss: 0.0661 - accuracy: 0.9724 - f1_m: 0.8610 - recall_m: 0.8395 - precision_m: 0.8837 - sensitivity_at_specificity: 0.9971 - specificity: 0.9779 - val_loss: 0.1087 - val_accuracy: 0.9601 - val_f1_m: 0.8004 - val_recall_m: 0.8437 - val_precision_m: 0.7616 - val_sensitivity_at_specificity: 0.9846 - val_specificity: 0.9646
Epoch 18/20
Epoch 00018: val_loss did not improve from 0.09229
180000/180000 [==============================] - 450s 2ms/sample - loss: 0.0678 - accuracy: 0.9719 - f1_m: 0.8581 - recall_m: 0.8354 - precision_m: 0.8824 - sensitivity_at_specificity: 0.9968 - specificity: 0.9774 - val_loss: 0.1037 - val_accuracy: 0.9613 - val_f1_m: 0.8037 - val_recall_m: 0.8364 - val_precision_m: 0.7736 - val_sensitivity_at_specificity: 0.9860 - val_specificity: 0.9662
Epoch 19/20
Epoch 00019: val_loss did not improve from 0.09229
180000/180000 [==============================] - 449s 2ms/sample - loss: 0.0651 - accuracy: 0.9728 - f1_m: 0.8632 - recall_m: 0.8426 - precision_m: 0.8850 - sensitivity_at_specificity: 0.9972 - specificity: 0.9782 - val_loss: 0.1082 - val_accuracy: 0.9600 - val_f1_m: 0.8000 - val_recall_m: 0.8442 - val_precision_m: 0.7604 - val_sensitivity_at_specificity: 0.9856 - val_specificity: 0.9644
Epoch 20/20
Epoch 00020: val_loss did not improve from 0.09229
180000/180000 [==============================] - 449s 2ms/sample - loss: 0.0639 - accuracy: 0.9732 - f1_m: 0.8654 - recall_m: 0.8456 - precision_m: 0.8864 - sensitivity_at_specificity: 0.9973 - specificity: 0.9785 - val_loss: 0.1113 - val_accuracy: 0.9598 - val_f1_m: 0.7996 - val_recall_m: 0.8466 - val_precision_m: 0.7579 - val_sensitivity_at_specificity: 0.9837 - val_specificity: 0.9648

ROC (AUC) altındaki alan değerlendirildiğinde 0.9772 değeri ile DRIVE datasetinde diğer çalışmalardan daha iyi performans elde ettiği gözlemlenmiştir.

Area under ROC curve: 0.9772384877684104

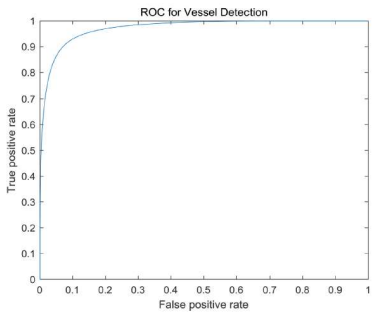


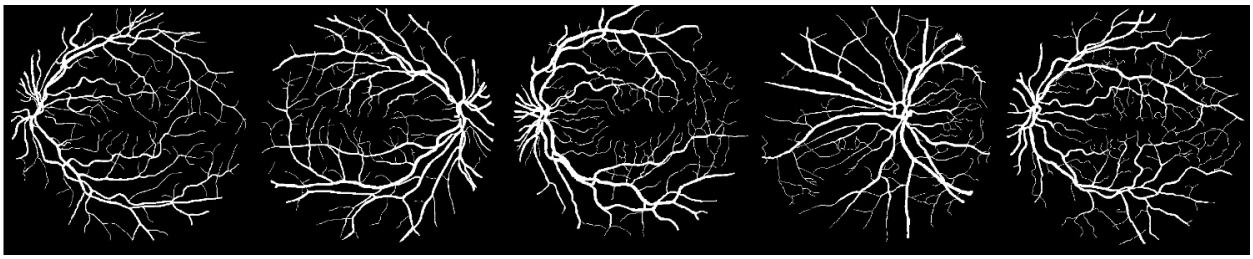

**Table 3**
Comparison with the state of art methods in DRIVE dataset.

| Method | AUC |
|---|---|
| Lahiri et al. [14] | 0.9500 |
| Maji et al. [12] | 0.9470 |
| Fu et al. [10] | 0.9523 |
| Soares et al. [2] | 0.9614 |
| Niemeijer et al. [15] | 0.9294 |
| Dasgupta et al. [9] | 0.9744 |
| Guo et al. [5] | 0.9476 |
| Sengur et al. [8] | 0.9674 |
| Azzopardi et al. [17] | 0.9614 |
| Osareh et al. [18] | 0.965 |
| Roychowdhury et al. [19] | 0.962 |
| Qiaoliang et al. [16] | 0.9738 |
| Proposed method | 0.9737 |

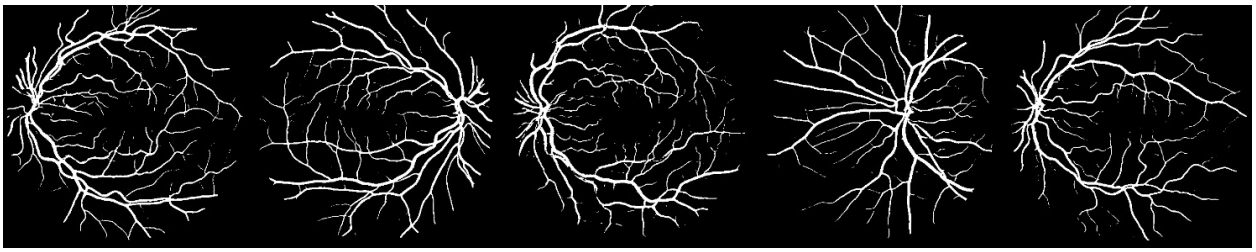
## Sonuç

Bu proje çalışmasında geliştirilmiş derin öğrenme U-Net modeline dayanan bir fundus retina damarları segmentasyonu gerçekleştirilmesi amaçlanmıştır.Drive Data seti üzerinde U-Net modeli ile fundus retina damarları segmentasyonu ile gerçekleştirilmiştir. Deneysel sonuçlar ile DRIVE veri setinde retinal damar segmentasyonu için doğruluk, duyarlılık ve özgüllük değerleri sırasıyla % 97,32,% 99,73 ve% 97,85 olarak elde edilmiştir. Deneysel sonuçlar diğer yöntemler ile karşılaştırılmış ve diğer çalışmalara göre ROC curve (AUC) altındaki alan değerlendirildiğinde 0.9772 AUC ile DRIVE datasetinde başarılı performans elde ettiği gözlemlenmiştir.

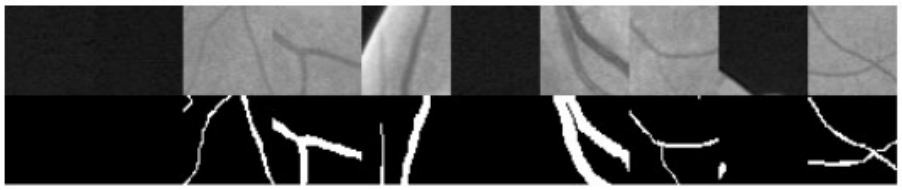
**Şekil 7** Orjinal görüntü



**Şekil 8** Ground Truth



**Şekil 9** U-NET model Segmentation Sonuçları



**Şekil 10** U-NET model Segmentation Sonuçları

198330404_Gamze_
Günay_U_NET_Mode

## Kaynaklar

[1] Jin, Qiangguo, et al. "DUNet: A deformable network for retinal vessel segmentation." Knowledge-Based Systems 178 (2019): 149-162.
[2] Marcos Ortega, M.G. Penedo, J. Rouco, N. Barreira, M.J. Carreira, "Personal verification based on extraction and characterization of retinal feature points."Journal of Visual Languages & Computing 20.2 :80-90, 2009.
[3] Simon and I. Goldstein. A new scientific method of identification. New York State Journal of Medicine, 35(18):901–906, Sept. 1935.
[4] Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234– 241. Springer, Cham, 2015.
[5] Wang B., Qiu S., He H. (2019) Dual Encoding U-Net for Retinal Vessel Segmentation. In: Shen D. et al. (eds) MICCAI 2019. Lecture Notes in Computer Science, vol 11764. Springer, Cham, 2019.
[6] Y Wu, Y Xia, Y Song, D Zhang, D Liu, C Zhang, W Cai. (2019) Vessel-Net: Retinal Vessel Segmentation Under Multi-path Supervision. Medical Image Computing and Computer Assisted Intervention – MICCAI 2019. Lecture Notes in Computer Science, vol 11764. Springer, Cham, 2019.
[7] Y. Guo, U.Budak, A. Sengür "A novel retinal vessel detection approach based onmultiple deep convolution neural networks," Comput. Methods Programs Biomed. ,Vol.167.,Pp. 43-48, 2018.
[8] P. Xiuqin, Q. Zhang, H. Zhang, And S. Li, ''A Fundus Retinal Vessels Segmentation Scheme Based On The Improved Deep Learning Unet Model,'' Ieee Access, Vol. 7, Pp. 122634–122643, 2019
[9] Z. Yan, X. Yang, And K.-T. T. Cheng, ''A Three-Stage Deep Learning Model For Accurate Retinal Vessel Segmentation,'' Ieee Journal Of Biomedical And Health Informatics, Vol. 23, No. 4, 2019
[10] https://drive.grand-challenge.org/