Teknoloji Fakültesi
Elektrik - Elektronik Mühendisliği Bölümü

DERİN SİNİR AĞLARI VE UYGULAMALARI DERSİ

U-Net MODEL RETİNA DAMARLARI SEGMENTASYONU

DERSİN ÖĞRETİM ÜYESİ

Prof. Dr. NECAATTİN BARIŞÇI

HAZIRLAYAN

Gamze GÜNAY

198330404

HAZİRAN 2020

U-Net Model Retina Damarları Segmentasyonu

PROBLEM TANIMI :

- Göz doktorunun retinayı taraması, hipertansiyon, kardiyovasküler ve diyabet gibi potansiyel göz hastalıklarının erken belirtilerini saptamasında önemli rol oynamaktadır.
- Retina kamera görüntülerinin manuel olarak değerlendirilmesi zordur ve çeşitli hastalardan çok sayıda görüntü alındığında manuel olarak değerlendirmesi olanaksız hale gelir.

ÇALIŞMANIN AMACI (Kapsam)

- 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.

U-Net Model Retina Damarları Segmentasyonu

DRIVE VERİ TABANI

- Digital Retinal Images for Vessel Extraction (DRIVE) veritabanı, 25-90 yaş aralığında 400 civarında diyabetik vakadan rastgele seçilen 40 adet renkli retinal fundus görüntüden oluşmaktadır. [1]
- 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 vaka içermektedir.

Eğitim Görüntüleri erken dönem diyabetik vaka içeren görüntüler

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

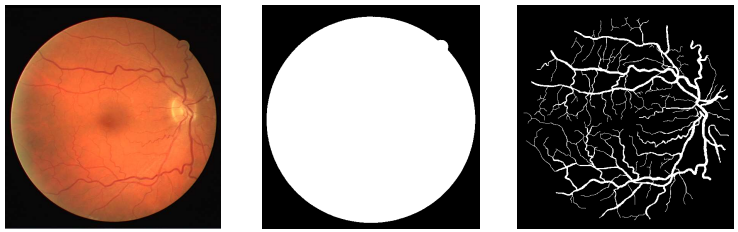
Test Görüntüleri erken dönem diyabetik vaka içeren görüntüler

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

U-Net Model Retina Damarları Segmentasyonu

DRIVE VERİ TABANI

- 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)mevcuttur.



Şekil 2 Orjinal görüntü/Maske Görüntüsü/Ground Truth

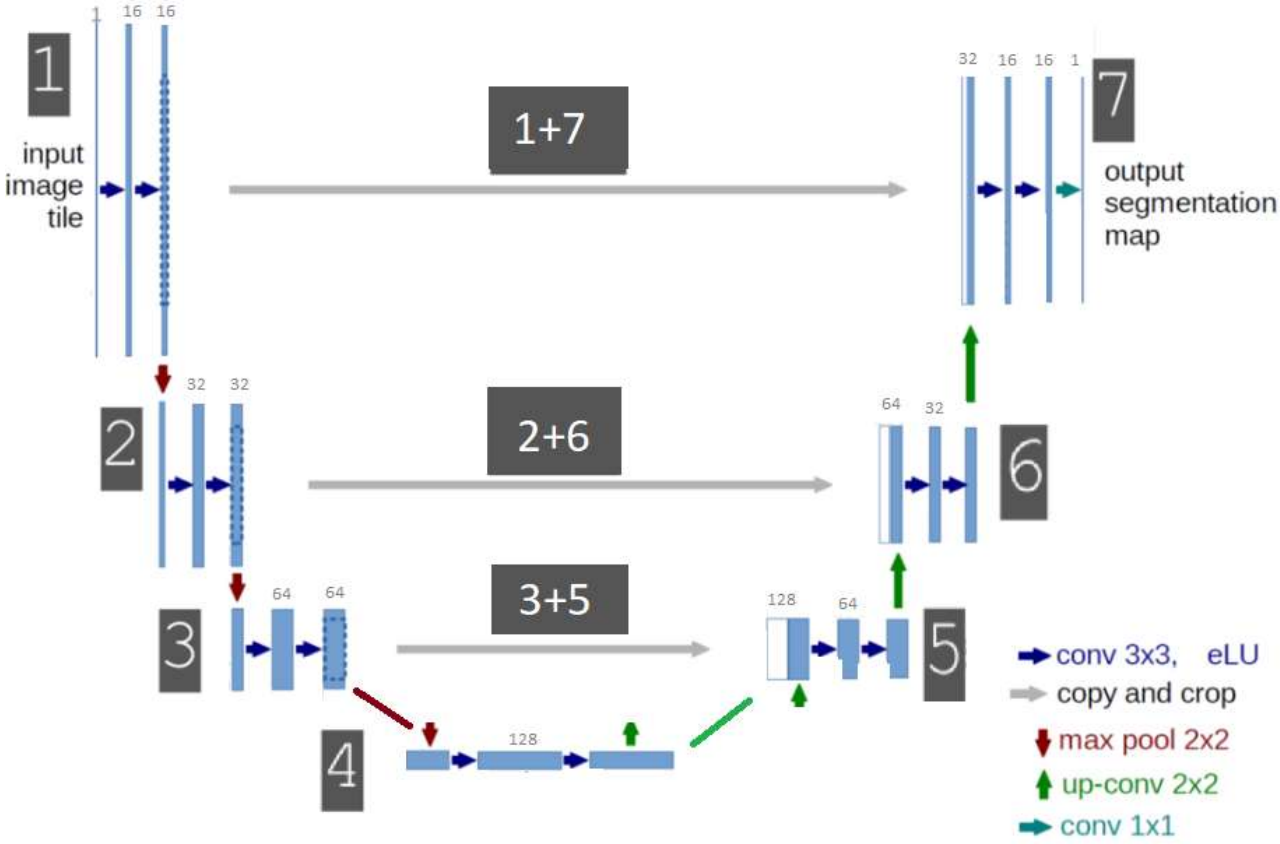
Pre-Processing:

- Derin öğrenme U-Net modelini öncesinde drive database verileri ön işleme tabi tutulmuştur. Sırasıyla:
  a) Orjinal DRIVE database den alınmış görüntü
  b) RGB to gray
  c) Normalization
  d) Histogram Equalization (CLAHE) görüntüde histogram eşitleme gerçekleştirilmiştir.
  e) Gamma correction işlemi yapılmıştır.

U-Net Model Retina Damarları Segmentasyonu

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.

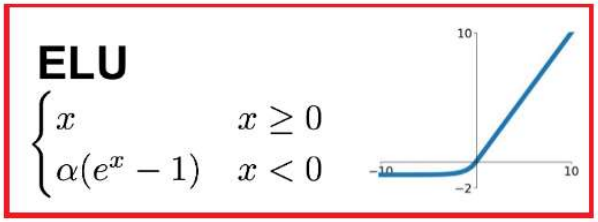


Şekil 2 Retina damar segmentasyonu için çalışmada kullanılan U-Net mimarisi

U-Net Model Retina Damarları Segmentasyonu

NETWORK YAPISI:

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



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

- U-NET mimarisinin kodlayıcı bölümüne convolution işlemleri sırası ile 16, 32,64 adet 3x3 filtre ile uygulanmıştır.

```
# 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 3 Retina damar segmentasyonu için çalışmada kullanılan U-Net mimarisi

U-Net Model Fundus Retina Damarları Segmentasyonu

NETWORK YAPISI:

- U-NET mimarisinin köprü bölümüne her bir convolution bloğun evrişimsel katmanı, eLU aktivasyon konksiyonundan geçirilir ve iki convolution blok arasında Dropout layer 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 Dropout 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 layer 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

U-Net Model Retina Damarları Segmentasyonu

AĞIRLIKLARIN GÜNCELLENMESİ :

- 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, epsilon 1e-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 FONKSİYONU:

- Kayıp fonksiyonunun hesaplanmasında ikili çapraz entropi (binary cross entropy) kullanılmıştır.

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

- 200000 adet patch (48x48x1) ile 20 epoch boyunca eğitim gerçekleştirilmiştir.
- Batch size 128 olarak uygulanmıştır.

U-Net Model Retina Damarları Segmentasyonu

Test Sonuçları

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

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

U-Net Model Retina Damarları Segmentasyonu

- 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

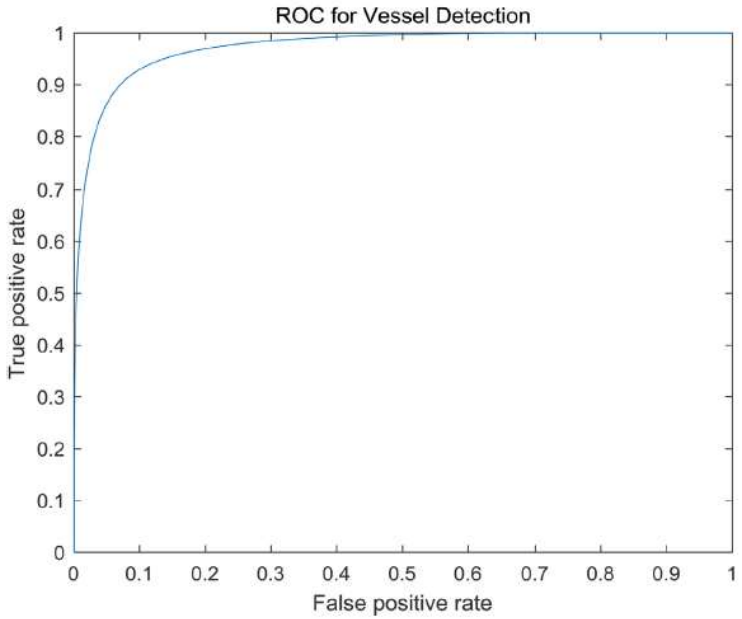


Table 3
Comparison with the state of art methods in DRIVE dataset.

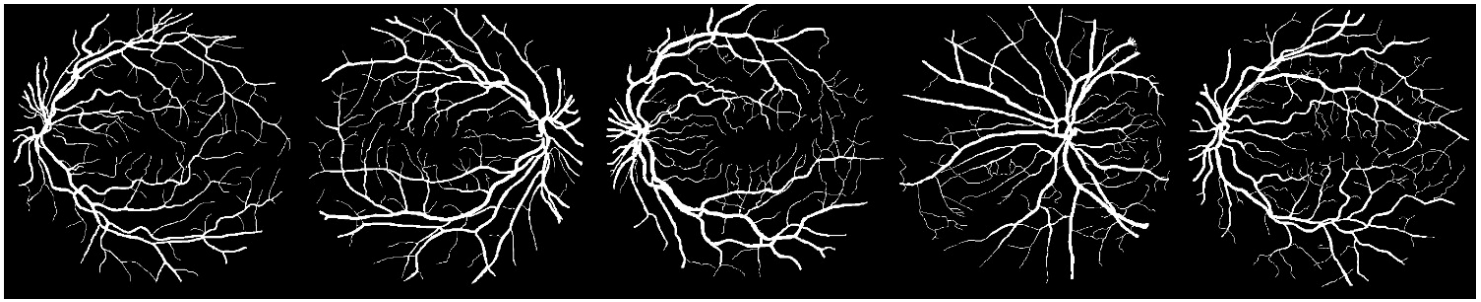
Method	AUC
Lahiri et al. [14]	0.9500
Maji et al. [12]	0.9470
Fu et al. [10]	0.9523
Soares et al. [2]	0.9614
Niemeijer et al. [15]	0.9294
Dasgupta et al. [9]	0.9744
Guo et al. [5]	0.9476
Sengur et al. [8]	0.9674
Azzopardi et al. [17]	0.9614
Osareh et al. [18]	0.965
Roychowdhury et al. [19]	0.962
Qiaoliang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

Şekil 6 ROC (AUC) altındaki alan ve Diğer çalışmaların karşılaştırılması

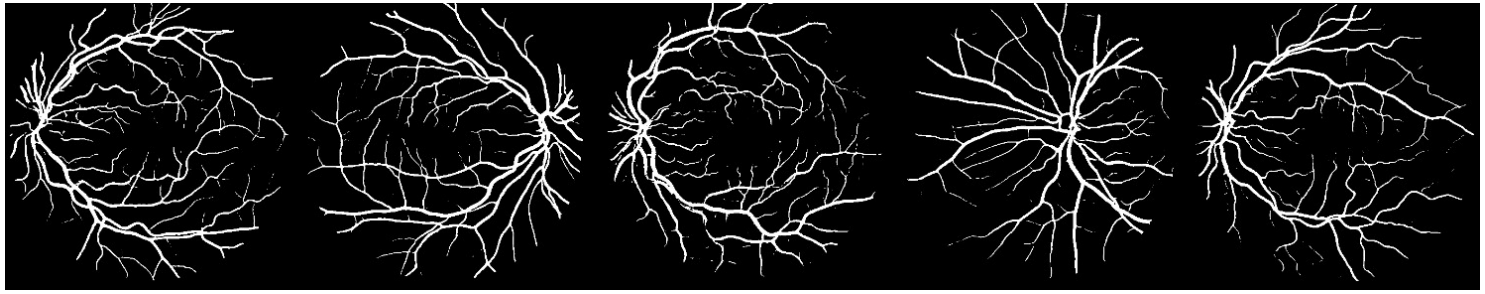
U-Net Model Retina Damarları Segmentasyonu

.

Şekil 7 Orjinal görüntü



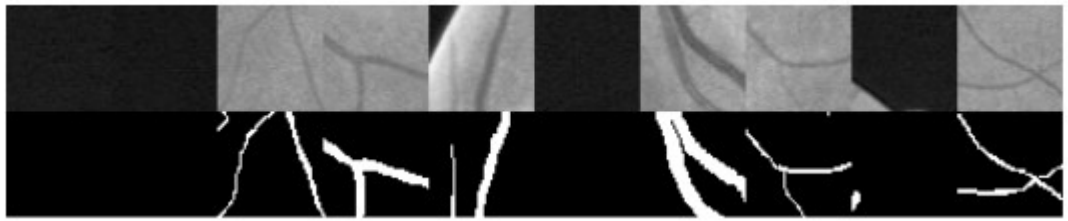
Şekil 8 Ground Truth



Şekil 9 U-NET model Segmentation sonuçları

U-Net Model Retina Damarları Segmentasyonu

.



Şekil 10 U-NET model Segmentation Sonuçları

KAYNAKLAR

[1] https://drive.grand-challenge.org/.

Beni Sunumum boyunca sabırla dinleyen siz değerli Arkadaşlarım
ve Sayın Necaattin BARIŞÇI Hocama,

TEŞEKKÜRLER

İLETİŞİM

gamze.gunay@ayesas.com