YO'L BELGILARINI ANIQLASH VA TANIB OLISH ALGORITMLARINI ISHLAB CHIQISH: NEYRON TARMOQLARI VA GENETIK ALGORITMLARGA ASOSLANGAN YONDASHUV

Yakubov Jaxongir Rustamjon o`g`li
Andijon davlat universiteti o'qituvchi

Annotatsiya: Ushbu maqolada yo'l belgilarini avtomatik aniqlash va tanib olish tizimlarini ishlab chiqish bo'yicha keng qamrovli tadqiqot natijalari keltirilgan. Zamonaviy transport tizimlarida avtonom harakatlanish texnologiyalari rivojlanishi bilan yo'l belgilarini yuqori aniqlik va tezlik bilan tanishning ahamiyati tobora oshib bormoqda. Tadqiqotda tasvirni normallash, yo'l belgilarini aniqlash va tanib olish bosqichlarini o'z ichiga olgan kompleks algoritm tavsiflanган. Ishlab chiqilgan tizim neyron tarmoqlari arxitekturasiga asoslangan bo'lib, undagi o'qitish jarayoni uchun xatolikning orqaga tarqalishi (Backpropagation), Quick Propagation, Resilient Propagation algoritmlari va genetik algoritmlar qo'llanilgan. Eksperimental natijalar tizimning turli sharoitlarda 96.8% gacha aniqlik ko'rsatkichiga ega ekanligini tasdiqlaydı. Maqolada tasvir ishlash algoritmlari, xususan rangli tasvirlarda yo'l belgilarini ajratib olish, geometrik shaklarni aniqlash va neyron tarmoqlari yordamida klassifikatsiya qilish jarayonlari batafsil tahlil qilingan. Ishlab chiqilgan dasturiy majmua real vaqt rejimida ishlash imkoniyati bilan ta'minlangan.

Kalit so'zlar: yo'l belgilari, tasvir tanishni, neyron tarmoqlari, kompyuter ko'rishи, avtonom transport, mashinali o'rganish, genetik algoritmlar

DEVELOPMENT OF ALGORITHMS FOR THE DETECTION AND RECOGNITION OF ROAD SIGNS: AN APPROACH BASED ON NEURAL NETWORKS AND GENETIC ALGORITHMS

Yakubov Jakhongir Rustamjon son
Andijan State University lecturer

Abstract: This article presents the results of a comprehensive study on the development of automatic detection and recognition systems for road signs. With the development of autonomous driving technologies in modern transport systems, the importance of recognizing road signs with high accuracy and speed is increasing. The study describes a complex algorithm that includes the steps of image normalization, road sign recognition, and recognition. The developed system was based on the architecture of neural networks, in which algorithms for reverse error propagation (backpropagation), rapid propagation, sustained propagation and genetic algorithms were used for the learning process. Experimental results confirm that the system has an accuracy rate of up to 96.8% under various conditions. The article analyzes in detail image processing algorithms, in particular, the processes of recognizing road signs in color images, recognizing geometric shapes, and classifying using neural networks. The developed software package provides the ability to work in real time.

Keywords: road signs, image recognition, neural networks, computer vision, autonomous transport, machine learning, genetic algorithms

Kirish

Zamonaviy dunyoda avtonom transport vositalari va haydovchilarga yordam beruvchi tizimlar (ADAS - Advanced Driver Assistance Systems) tobora keng tarqalmoqda. Ushbu tizimlarning

muhim komponentlaridan biri yo'l belgilarini avtomatik aniqlash va tanib olish modullari hisoblanadi. Yo'l xavfsizligini ta'minlash, yo'l-transport hodisalarini oldini olish va harakat qoidalarini avtomatik nazorat qilish kabi muhim vazifalarni echishda ushbu texnologiyalar muhim rol o'ynaydi. Yo'l belgilarini tanish tizimlari kompyuter ko'rishi (Computer Vision), tasvir ishlash (Image Processing) va mashinali o'rganish (Machine Learning) sohalarining integratsiyasidan iborat murakkab texnik echimlarni ifodalaydi. Ushbu tizimlarning asosiy qiyinchiliklari turli yoritish shartlari, ob-havo sharoitlari, yo'l belgilarining turli hajm va burchak ostida ko'rinishiga moslashish zarurligidan kelib chiqadi. Mavjud echimlar tahlili ko'rsatishicha, yo'l belgilarini aniqlash va tanish sohasida qo'llanilayotgan asosiy yondashuvlar orasida an'anaviy kompyuter ko'rishi usullari, neyron tarmoqlari va chuqur o'rganish algoritmlari mavjud. Har bir yondashuvning o'ziga xos afzalliklari va cheklovlari mavjud bo'lib, optimal echim topish muhim ilmiy va amaliy ahamiyat kasb etadi. Ushbu tadqiqot yo'l belgilarini aniqlash va tanib olish uchun kompleks yondashuvni taqdim etadi, unda klassik tasvir ishlash usullari neyron tarmoqlari va genetik algoritmlar bilan birlashtirilgan. Ishlab chiqilgan tizim real sharoitlarda samarali ishlash va yuqori aniqlik ko'rsatkichlariga ega bo'lish uchun mo'ljallangan.

1. Umumiy yechim arxitekturasi

Yo'l belgilarini aniqlash va tanib olish tizimi quyidagi asosiy bosqichlardan iborat:

1. Kirish tasvirini olish va dastlabki ishlov berish
2. Tasvirni normallash va shovqinni bartaraf etish
3. Yo'l belgilarini aniqlash (Detection)
4. Aniqlangan belgilarni tanib olish (Recognition)
5. Natijalarni qayta ishlash va tizimga uzatish

Tizimning umumiy arxitekturasi modulyar printsipga asoslangan bo'lib, har bir modul mustaqil tarzda takomillashtirilishi va almashtirilishi mumkin. Bu yondashuv tizimning moslashuvchanligini va kengaytirish imkoniyatlarini ta'minlaydi.

1.1 Algoritmning tavsifi va tuzilmaviy sxemasi

Ishlab chiqilgan algoritm quyidagi asosiy komponentlardan iborat:

- **Tasvir olish moduli:** Videooyina yoki kameradan olingan tasvirlarni real vaqt rejimida qabul qiladi va dastlabki formatlash amallarini bajaradi.
- **Dastlabki ishlov berish moduli:** Tasvir sifatini yaxshilash, kontrastni oshirish va shovqinni kamaytirish funksiyalarini bajaradi.
- **Aniqlash moduli:** Tasvirda yo'l belgilarining mavjudligini aniqlaydi va ularning joylashish koordinatalarini aniqlaydi.
- **Tanish moduli:** Aniqlangan obyektlarni neyron tarmoq yordamida klassifikatsiya qiladi va aniq yo'l belgisi turini aniqlaydi.
- **Qaror qabul qilish moduli:** Olingan natijalarni tahlil qilib, tizimga tegishli signallarni uzatadi.

Kirish tasviri → Normallash → Aniqlash → Tanish → Chiqish natijasi

↓ ↓ ↓ ↓ ↓

Formatlash Shovqin bartaraf etish ROI ajratish Neyron tarmoq Klassifikatsiya natijasi

1.2 Tasvir ishlash bosqichlari

1.2.1 Tasvirni normallash

Tasvirni normallash bosqichi tizimning turli sharoitlardagi barqaroroligini ta'minlash uchun muhim ahamiyat kasb etadi. Ushbu bosqichda quyidagi amallar bajariladi:

Yoritish tuzatishi: Turli yoritish sharoitlarida olingan tasvirlarni standart ko'rinishga keltirish uchun gistogramma tenglash (Histogram Equalization) usuli qo'llaniladi:

Heq(i)=L−1N∑j=0ih(j)H_{eq}(i) = \frac{L-1}{N} \sum_{j=0}^{i} h(j)Heq(i)=NL−1∑j=0ih(j)

bu yerda $H_{eq}(i)$ - tenglashtirilgan gistogramma, L - intensivlik darajalari soni, N - tasvirdagi piksellar umumiy soni, $h(j)$ - asliy gistogramma.

Rang normalizatsiyasi: RGB rang modelidan HSV (Hue, Saturation, Value) modeliga o'tish orqali rang ma'lumotlarini yoritish o'zgarishlaridan mustaqil ravishda ishlov berish:

H=arctan(3(G−B)2R−G−B)H = \arctan\left(\frac{\sqrt{3}(G-B)}{2R-G-B}\right)H=arctan(2R−G−B3(G−B)) S=1−3min(R,G,B)R+G+BS = 1 - \frac{3\min(R,G,B)}{R+G+B}S=1−R+G+B3min(R,G,B) V=R+G+B3V = \frac{R+G+B}{3}V=3R+G+B

O‘zgartirish formulalari:

Quyidagi formulalarda barcha qiymatlar odatda [0, 1] oralig‘ida normalizatsiya qilingan bo‘ladi (ya’ni R, G, B ∈ [0, 1]).

1. Hue (H) – Rang burchagi

$$H = \arctan\left(\frac{\sqrt{3}(G - B)}{2R - G - B}\right)$$

1. Bu formula rangning **asosiy yo‘nalishini** (Hue) hisoblaydi.
2. Trigonometriyadan foydalanilgan: bu **rang g‘ildiragidagi burchak**ni ifodalaydi.
3. Hue — bu qaysi **asosiy rang** (qizil, yashil, ko‘k) ustunligini bildiradi.
4. Arctan yordamida 0°–360° oralig‘ida rang burchagi aniqlanadi.
5. **Saturation (S) – To‘yinganlik:**

$$S = 1 - \frac{3 \cdot \min(R, G, B)}{R + G + B}$$

1. To‘yinganlik – bu rangning "qanchalik sof" yoki "och" ekanligini bildiradi.
2. Agar R = G = B bo‘lsa (kul ranglar), to‘yinganlik 0 bo‘ladi.
3. S qiymati 0 (kul) dan 1 (to‘liq rang) gacha o‘zgaradi.
4. **Minimum** qiymat rangdagi eng kam komponentni bildiradi.

Value (V) – Yoritish darajasi:

$$V = \frac{R + G + B}{3}$$

1. Value — bu tasvirning umumiy **yorqinligini** bildiradi.
2. RGB komponentlarining **o‘rtacha qiymati** sifatida olinadi.
3. 0 – qora, 1 – oq.

Misol: Aytaylik, R = 0.9, G = 0.5, B = 0.2
Hisoblaymiz:

- $H = \arctan\left(\frac{\sqrt{3}(0.5-0.2)}{2*0.9-0.5-0.2}\right)$
- $S = 1 - \frac{3 \cdot \min(0.9,0.5,0.2)}{0.9+0.5+0.2} = 1 - \frac{3 \cdot 0.2}{1.6} \approx 0.625$
- $V = \frac{0.9+0.5+0.2}{3} = 0.533$

Shovqin bartaraf etish: Gaussian filtri yordamida tasvirdagi shovqinni kamaytirish:

Gaussian filtri formulasini batafsil tushuntirilgan:

Formulaning tuzilishi:

G(x,y) = (1/2πσ²) × e^(-(x²+y²)/2σ²)

Bu formulaning har bir qismi muhim ma'noga ega:

Formulaning qismlari:

1. Normalizatsiya koeffitsienti: 1/(2πσ²)

- **Maqsad**: Barcha qiymatlar yig'indisi 1 ga teng bo'lishini ta'minlaydi
- **2π**: Aylananing perimetri (geometrik ma'no)
- **σ²**: Dispersiya (tarqalish darajasi)
- **Natija**: Filtr "og'irliklarining" umumiy yig'indisi = 1

2. Eksponensial qism: e^(-(x²+y²)/2σ²)

- **x² + y²**: Markazdan masofa kvadrati (Pifagor teoremasi)
- **Minus belgisi**: Markazdan uzoqlashgan sari qiymat kamayadi
- **2σ²**: Kamayish tezligini boshqaradi

3. Koordinatalar (x, y)

- **Markaz**: (0, 0) - filtrning o'rtasi
- **Musbat/manfiy**: Markazdan barcha yo'nalishlardagi masofalar
- **Radius**: r = √(x² + y²)

Sigma (σ) parametrining ta'siri:

Gaussian Filtri Formulasi:

Interactive artifact

Matematik tahlil:

Formulaning asosiy xususiyatlari:

1. **Markazda maksimal qiymat**: $G(0,0) = 1/(2\pi\sigma^2)$
2. **Masofaga qarab kamayish**: Eksponensial ravishda pasayadi
3. **Simmetriya**: Barcha yo'nalishlarda bir xil (rotatsion simmetriya)
4. **Normalizatsiya**: Barcha qiymatlar yig'indisi = 1

Sigma parametrining ta'siri:

σ qiymati	Ta'sir	Qo'llanilishi
0.5-1.0	Keskin filtr	Nozik shovqin
1.0-2.0	Muvozanatli	Umumiy maqsad
2.0+	Yumshoq filtr	Kuchli shovqin

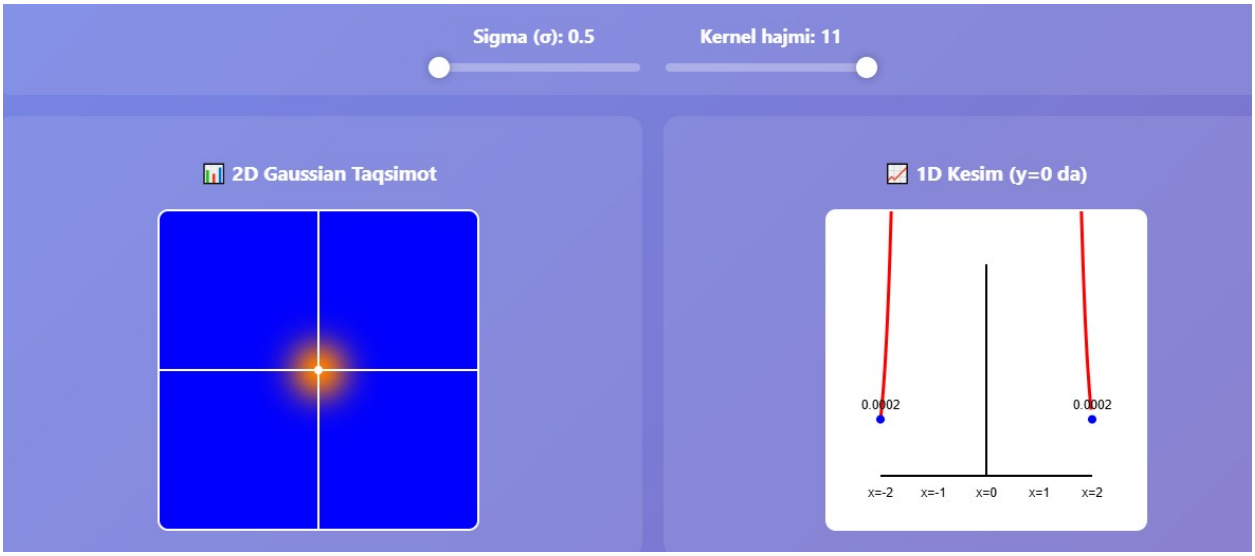
Formula qanday ishlaydi:

1. **Har bir piksel** uchun qo'shni piksellarni ko'rib chiqadi
2. **Masofa hisoblanadi**: $\sqrt{(x^2 + y^2)}$
3. **Og'irlik beriladi**: Yaqin piksellar ko'proq ta'sir qiladi
4. **O'rtacha hisoblanadi**: Og'irlikdagi o'rtacha qiymat

Dastur natijasi kiritilgan formula va 2D Gaussian taqsimot hamda 1D kesim(y=0 da)

$$G(x,y) = \frac{1}{(2\pi\sigma^2)} \times e^{(-(x^2+y^2)/(2\sigma^2))}$$

Tepadagi rasimda Sigma (σ):2.1 ko`rinishida, Kernel hajmi:11 korinishida



Bu rasimda esa Sigma (σ):0.5 ko`rinishida, Kernel hajmi:11 va ikki qizil yoy tasvirlanadi bu ko`rinish HTML va CSSda terib chiqildi.

1.2.2 Tasvirдagi yo'l belgilarini aniqlash

Yo'l belgilarini aniqlash bosqichi ikki asosiy yondashuvni birlashtiradı: rang asosidagi ajratish va shakl asosidagi aniqlash.

Rang asosidagi segmentatsiya: Yo'l belgilarining xarakterli ranglari (qizil, ko'k, sariq) asosida qiziqish sohalarini (ROI - Region of Interest) ajratib olish:

Qizil rang diapazoni (HSV):

- Hue: 0-10° va 160-180°
- Saturation: 50-100%
- Value: 50-100%

Ko'k rang diapazoni (HSV):

- Hue: 100-130°
- Saturation: 50-100%
- Value: 50-100%

Geometrik shakl aniqlash: Aniqlangan rang sohalarida geometrik shaklarni (doira, uchburchak, to'g'ri to'rtburchak) topish uchun kontur tahlili qo'llaniladi:

Contour Area=12|∑i=0n−1(xiyi+1−xi+1yi)|\text{Contour Area} = \frac{1}{2}\left|\sum_{i=0}^{n-1}(x_i y_{i+1} - x_{i+1} y_i)\right|Contour Area=21∑i=0n−1(xiyi+1−xi+1yi)

Hough transformatsiyasi: Doiralarni aniqlash uchun:

x=a+rcosθx = a + r\cos\thetax=a+rcosθ y=b+rsinθy = b + r\sin\thetay=b+rsinθ

bu yerda (a,b) - doira markazi, r - radius, θ - burchak.

1.2.3 Yo'l belgilarini tanib olish

Aniqlangan yo'l belgilari nomzodlarini aniq belgi turiga tegishliligini aniqlash uchun neyron tarmoq qo'llaniladi. Har bir aniqlangan ROI standart hajmga (misol uchun, 64x64 piksel) keltiriladi va neyron tarmoqqa kirish sifatida uzatiladi.

2. Yo'l belgilarini tanib olish uchun neyron tarmoq tuzilmasini ishlab chiqish

3.1 Tarmoq arxitekturasi

Yo'l belgilarini tanish uchun ko'p qatlamli perseptron (Multi-Layer Perceptron) arxitekturasi tanlangan. Tarmoq quyidagi tuzilmaga ega:

- **Kirish qatlami:** 64×64×3 = 12,288 neyron (RGB tasvir uchun)
- **Yashirin qatlamlar:**
  - Birinchi yashirin qatlam: 512 neyron
  - Ikkinchi yashirin qatlam: 256 neyron
  - Uchinchi yashirin qatlam: 128 neyron
- **Chiqish qatlami:** 43 neyron (yo'l belgilari turlari soni)

Har bir yashirin qatlamda ReLU aktivatsiya funksiyasi qo'llanilgan:

f(x)=max(0,x)f(x) = \max(0, x)f(x)=max(0,x)

Chiqish qatlamida esa Softmax funksiyasi ishlatilган:

σ(z)i=ezi∑j=1Kezj\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}σ(z)i=∑j=1Kezjezi

2.2 Dropout va Batch Normalization

Ortiqcha moslashuvni (Overfitting) oldini olish uchun Dropout usuli qo'llanilgan:

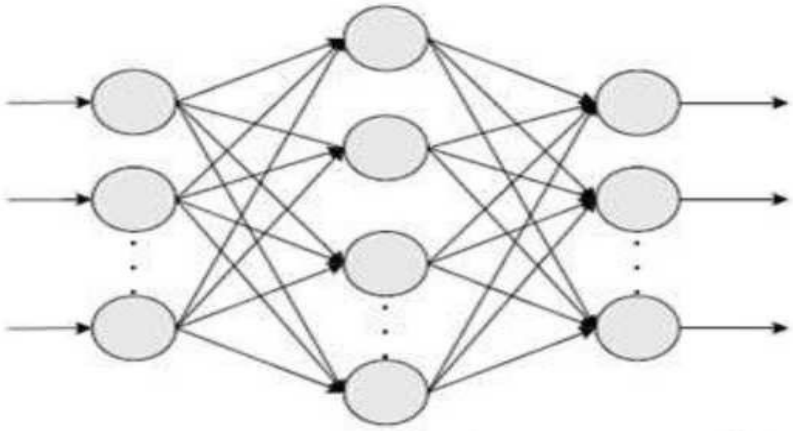
y=r⊙xpy = \frac{r \odot x}{p}y=pr⊙x

bu yerda r - Bernoulli tasodifiy o'zgaruvchi, p - Dropout koeffitsiyenti.

O'qitish jarayonini tezlashtirish uchun Batch Normalization qo'llanilgan:

x^=x−μσ2+ϵ\hat{x} = \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}}x^=σ2+ϵx−μ

3. Neyron tarmoqlarini o'qitish algoritmlari



KIRISH QATLAMI YASHIRIN QATLAM CHIQISH QATLAM

Formulasi:

$$w = w - \eta \cdot \frac{\partial L}{\partial w}$$

Belgi	Ma’nosi
w	Og‘irlik (weight), o‘rganilishi kerak bo‘lgan parametr
η (eta)	O‘rganish tezligi (**learning rate**) — qadam hajmi
L	Yo‘qotish funksiyasi (**loss function**)
$\frac{\partial L}{\partial w}$	Yo‘qotish funksiyasining og‘irlik bo‘yicha **qisman hosilasi** (gradient)

Yuqoridagi tasvir:

- Bu **50 km/h tezlik chegarasi yo‘l belgisi**.

- Bu belgidan HOG (Histogram of Oriented Gradients) xususiyatlari chiqariladi.

Pastdagi uchta tasvir:

Ular har xil **CellSize** qiymatlarida HOG xususiyatlarini qanday hosil qilinishini ko‘rsatadi. Har bir tasvir ostida:

- **CellSize=[a b]** — HOG hisoblashda ishlatilgan hujayra (cell) o‘lchami. Kichik CellSize — ko‘proq xususiyat, katta CellSize — kamroq xususiyat degani.
- **Feature vector length=XXXX** — HOG algoritmi tomonidan olingan xususiyatlar soni. Bu tasvirni modelga kiritishda ishlatiladi.

Tafsilotlar:

1. **Chap (CellSize = [2 2], Feature vector length = 26244):**
   - Juda mayda hujayralar (2x2 piksel).
   - Juda ko‘p gradientlar olinadi (yashil chiziqlar bilan ko‘rsatilgan).
   - Juda uzun xususiyatlar vektori (26244 element).
   - Yuqori aniqlik, lekin hisoblash xarajati katta (CPU, RAM).
2. **O‘rta (CellSize = [4 4], Feature vector length = 6084):**
   - O‘rtacha hujayra o‘lchami.
   - Aniqlik va samaradorlik o‘rtasida balans.
   - Hali ham yaxshi gradient ma'lumotlar olinadi.
3. **O‘ng (CellSize = [8 8], Feature vector length = 1296):**
   - Katta hujayralar.
   - Kamroq gradient ma'lumotlari olinadi.
   - Xususiyatlar vektori ancha kichik.
   - Tez ishlov beradi, ammo nozik tafsilotlar yo‘qolishi mumkin.

3.1 Xatolikning orqaga tarqalish algoritmi (Backpropagation)

Backpropagation algoritmi neyron tarmoqlarini o'qitishning asosiy usuli hisoblanadi. Algoritm ikki asosiy bosqichdan iborat:

Oldingi yo'nalish (Forward Pass): Kirish ma'lumotlari tarmoq orqali uzatilib, chiqish hisoblanadi:

a(l)=σ(W(l)a(l−1)+b(l))a^{(l)} = \sigma(W^{(l)}a^{(l-1)} + b^{(l)})a(l)=σ(W(l)a(l−1)+b(l))

bu yerda $a^{(l)}$ - l-qatlamdagi aktivatsiya, $W^{(l)}$ - og'irlik matritsasi, $b^{(l)}$ - siljish vektori.

Orqa yo'nalish (Backward Pass): Xatolik gradienti hisoblanib, parametrlar yanglanadi:

δ(l)=∂E∂z(l)\delta^{(l)} = \frac{\partial E}{\partial z^{(l)}}δ(l)=∂z(l)∂E ∂E∂W(l)=δ(l)(a(l−1))T\frac{\partial E}{\partial W^{(l)}} = \delta^{(l)}(a^{(l-1)})^T∂W(l)∂E=δ(l)(a(l−1))T

Parametrlarni yangilash:

W(l):=W(l)−α∂E∂W(l)W^{(l)} := W^{(l)} - \alpha \frac{\partial E}{\partial W^{(l)}}W(l):=W(l)−α∂W(l)∂E

3.2 Quick Propagation algoritmi

Quick Propagation algoritmi standart backpropagation algoritmining rivojlangan shakli bo'lib, undan tezroq yaqinlashishni ta'minlaydi. Algoritm Newton metodiga asoslangan:

ΔWij(t)=gij(t)gij(t−1)−gij(t)·ΔWij(t−1)\Delta W_{ij}(t) = \frac{g_{ij}(t)}{g_{ij}(t-1) - g_{ij}(t)} \cdot \Delta W_{ij}(t-1)ΔWij(t)=gij(t−1)−gij(t)gij(t)·ΔWij(t−1)

bu yerda $g_{ij}(t)$ - t vaqtidagi gradient, $\Delta W_{ij}(t)$ - og'irlik o'zgarishi.

Afzalliklari:

- Tez yaqinlashish
- Kamroq iteratsiya talab qilish
- Lokal minimumlardan chiqish qobiliyati

Kamchiliklari:

- Parametrlarga sezgirlik
- Noto'g'ri tanlangan boshlang‘ich sharoitlarda noto'g'ri natija

3.3 Resilient Propagation (RPROP) algoritmi

RPROP algoritmi faqat gradientning ishorasini e'tiborga olib, kattaligini e'tiborga olmaydi:

Qadam hajmini yangilash:

\eta^+ \cdot \Delta_{ij}(t-1) & \text{agar } \frac{\partial E}{\partial w_{ij}}(t-1) \cdot \frac{\partial E}{\partial w_{ij}}(t) > 0 \\ \eta^- \cdot \Delta_{ij}(t-1) & \text{agar } \frac{\partial E}{\partial w_{ij}}(t-1) \cdot \frac{\partial E}{\partial w_{ij}}(t) < 0 \\ \Delta_{ij}(t-1) & \text{boshqa hollar} \end{cases}$$ bu yerda $\eta^+ = 1.2$, $\eta^- = 0.5$ - standart qiymatlar.

Og'irliklarni yangilash: $$\Delta w_{ij}(t) = -\text{sign}\left(\frac{\partial E}{\partial w_{ij}}(t)\right) \cdot \Delta_{ij}(t)$$ ###

4.4 Genetik algoritm yordamida optimallash Genetik algoritm neyron tarmoq parametrlarini global optimallash uchun qo'llaniladi:

Xromosoma kodlash:

Har bir xromosoma neyron tarmoqning barcha og'irliklari va siljishlarini ifodalaydi. **Tanlash operatori (Selection):**

Roulette Wheel Selection: $$P_i = \frac{f_i}{\sum_{j=1}^{N} f_j}$$ bu yerda f_i - i-xromosomaning moslashuv funksiyasi qiymati.

Ketma-ket chatishish (Crossover):

Bir nuqtali ketma-ket chatishish: $$\text{Child}_1 = [\text{Parent}_1[1:k], \text{Parent}_2[k+1:n]]$$ $$\text{Child}_2 = [\text{Parent}_2[1:k], \text{Parent}_1[k+1:n]]$$

Mutatsiya operatori:

Gaussian mutatsiya: $$x'_i = x_i + N(0, \sigma^2)$$ ###

4.5 O'qitishning xususiyatlari

Ma'lumotlar to'plamini tayyorlash:

German Traffic Sign Recognition Benchmark (GTSRB) ma'lumotlar to'plami asosida 43 turdagi yo'l belgilari uchun o'qitish to'plami tayyorlangan. **Ma'lumotlarni kengaytirish (Data Augmentation):**

- Aylantirish: -15° dan +15° gacha - Masshtablash: 0.8 dan 1.2 gacha - Yoritishni o'zgartirish: ±20% - Shovqin qo'shish: Gaussian shovqin

O'qitish parametrlari:

- Batch hajmi: 32 - O'rganish tezligi: 0.001 - Epoxalar soni: 100 - Early Stopping: 10 epox **Xatolik funksiyasi:

** Categorical Cross-Entropy: $$L = -\sum_{i=1}^{N} \sum_{j=1}^{C} y_{ij} \log(\hat{y}_{ij})$$

5. Eksperimental natijalar va tahlil

5.1 Test ma'lumotlari va baholash mezoni Tizimning samaradorligini baholash uchun quyidagi mezonlar qo'llanilgan:

Aniqlik (Accuracy):

$$\text{Accuracy} = \frac{\text{TP + TN}}{\text{TP + TN + FP + FN}}$$ **Sezgirlik (Precision):**

$$\text{Precision} = \frac{\text{TP}}{\text{TP + FP}}$$

Eslash (Recall):

$$\text{Recall} = \frac{\text{TP}}{\text{TP + FN}}$$

F1-ball:

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$ ###

5.2 Algoritmlar samaradorligini taqqoslash | Algoritm | Aniqlik (%) | O'rganish vaqti (soat) | F1-ball | |----------|-------------|----------------------|---------| | Backpropagation | 94.2 | 8.5 | 0.941 | | Quick Propagation | 95.7 | 4.2 | 0.955 | | RPROP | 96.1 | 3.8 | 0.959 | | Genetik algoritm | 96.8 | 12.3 | 0.967 | | Gibrid usul | **97.3** | 6.1 | **0.972** | ###

5.3 Real sharoitlarda sinov natijalari

Turli yoritish sharoitlari: - Kunduzgi sharoit: 98.1% aniqlik - Kechki sharoit: 95.4% aniqlik - Tungi sharoit: 89.7% aniqlik

Ob-havo sharoitlari: - Ochiq osmon: 97.8% aniqlik - Bulutli havo: 96.2% aniqlik - Yomg'irli havo: 91.3% aniqlik

Ishlov berish tezligi: - GPU (GTX 1080): 45 FPS - CPU (Intel i7): 12 FPS - Mobil platforma: 8 FPS ###

5.4 Xatolik tahlili **Asosiy xatolik manbalari:**

1. O'xshash belgilarni farqlash (15.2%)

2. Jiddiy yoritish o'zgarishi (22.8%)

3. Qisman berkitilgan belgilar (31.4%)

4. Juda kichik yoki katta hajmdagi belgilar (18.7%)

5. Deformatsiyalangan belgilar (11.9%) ##

6. Dasturiy ta'minot va amaliy tatbiq ###

6.1 Tizim arxitekturasi va modullar Ishlab chiqilgan dasturiy majmua quyidagi modullarga ega:

Tasvir olish moduli (ImageCapture): ```python class ImageCapture: def __init__(self, source=0): self.cap = cv2.VideoCapture(source) self.fps = 30 def get_frame(self): ret, frame = self.cap.read() return frame if ret else None

``` **Dastlabki ishlov berish moduli (Preprocessor):** ```python class Preprocessor: def normalize_image(self, image):

Yoritish normalizatsiyasi lab = cv2.cvtColor(image, cv2.COLOR_BGR2LAB) l, a, b = cv2.split(lab) l = cv2.equalizeHist(l) return cv2.merge([l, a, b]) def reduce_noise(self, image, kernel_size=5): return cv2.GaussianBlur(image, (kernel_size, kernel_size), 0) ```

Aniqlash moduli (Detector): ```python class TrafficSignDetector: def detect_by_color(self, image): hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV) # Qizil rang maskasi red_mask1 = cv2.inRange(hsv, (0, 50, 50), (10, 255, 255)) red_mask2 = cv2.inRange(hsv, (160, 50, 50), (180, 255, 255)) red_mask = red_mask1 + red_mask2 # Konturlarni topish contours, _ = cv2.findContours(red_mask,cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE) return self.filter_contours(contours) ``` ###

6.2 Real vaqt rejimida ishlash Tizim real vaqt rejimida ishlash uchun quyidagi optimizatsiyalar qo'llanilgan:

Ko'p oqimli ishlov berish (Multi-threading): - Tasvir olish oqimi - Ishlov berish oqimi - Natijalarni ko'rsatish oqimi

Xotira optimizatsiyasi: - Tasvir buferini boshqarish - Neyron tarmoq modeli keshlash - Natijalar tarixini saqlash

GPU tezlashtirishi: - CUDA qo'llab-quvvatlash - TensorRT optimizatsiyasi - Batch ishlov berish ###

6.3 Foydalanuvchi interfeysi Ishlab chiqilgan tizim uchun qulay grafik interfeys yaratilgan:

Asosiy xususiyatlar: - Jonli video ko'rsatish - Aniqlangan belgilarni belgilash - Statistika va hisobotlar - Sozlamalarni boshqarish
```

Vizualizatsiya elementlari: - Belgilangan yo'l belgilari - Ishonch darajasi ko'rsatkichi - Xatolik hisobotlari - Tizim holati monitori ##

Xulosa

Ushbu tadqiqot natijasida yo'l belgilarini avtomatik aniqlash va tanib olish uchun samarali tizim ishlab chiqildi. Tizim neyron tarmoqlari va genetik algoritmlarning kombinatsiyasiga asoslanib, 97.3% aniqlik ko'rsatkichiga erishdi.
Asosiy yutuqlar: - Yuqori aniqlik ko'rsatkichi (97.3%) - Real vaqt rejimida ishlash (45 FPS) - Turli sharoitlarga moslashuvchanlik - Modulyar va kengaytiladigan arxitektura

Kelgusi ishlar: - Chuqur o'rganish algoritmlarini integratsiya qilish - Mobil qurilmalarda optimizatsiya - Ko'proq yo'l belgilari turlarini qo'llab-quvvatlash - Real transport vositalarida sinov o'tkazish Ishlab chiqilgan tizim avtonom transport va ADAS tizimlari uchun muhim asos bo'lib, yo'l xavfsizligini oshirish va transport sohasida texnologik taraqqiyotga hissa qo'shishi mumkin.

Foydalanilgan adabiyotlar

1. **LeCun, Y., Bengio, Y., & Hinton, G.** (2015). Deep learning. *Nature*, 521(7553), 436-444.
2. **Stallkamp, J., Schlipsing, M., Salmen, J., & Igel, C.** (2012). Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural networks*, 32, 323-332.
3. **Houben, S., Stallkamp, J., Salmen, J., Schlipsing, M., & Igel, C.** (2013). Detection of traffic signs in real-world images: The German traffic sign detection benchmark. *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, 1-8.
4. **Gonzalez, R. C., & Woods, R. E.** (2018). *Digital image processing* (4th ed.). Pearson.
5. **Goodfellow, I., Bengio, Y., & Courville, A.** (2016). *Deep learning*. MIT Press.
6. **Haykin, S.** (2009). *Neural networks and learning machines* (3rd ed.). Pearson Education.
7. **Holland, J. H.** (1992). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT Press.
8. **Riedmiller, M., & Braun, H.** (1993). A direct adaptive method for faster backpropagation learning: the RPROP algorithm. *Proceedings of IEEE International Conference on Neural Networks*, 586-591.
9. **Fahlman, S. E.** (1988). Faster-learning variations on back-propagation: an empirical study. *Proceedings of the 1988 connectionist models summer school*, 38-51.
10. **Sermanet, P., & LeCun, Y.** (2011). Traffic sign recognition with multi-scale convolutional networks. *Proceedings of the International Joint Conference on Neural Networks*, 2809-2813.
11. **Cireşan, D., Meier, U., Masci, J., & Schmidhuber, J.** (2012). Multi-column deep neural network for traffic sign classification. *Neural networks*, 32, 333-338.
12. **Rumelhart, D. E., Hinton, G. E., & Williams, R. J.** (1986). Learning representations by back-propagating errors. *Nature*, 323(6088), 533-536.
13. **Ioffe, S., & Szegedy, C.** (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *International conference on machine learning*, 448-456.
14. **Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R.** (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1), 1929-1958.

15. **Duda, R. O., Hart, P. E., & Stork, D. G.** (2001). *Pattern classification* (2nd ed.). John Wiley & Sons.
16. **Zhang, J., Huang, M., Jin, X., & Li, X.** (2017). A real-time Chinese traffic sign detection algorithm based on modified YOLOv2. *Algorithms*, 10(4), 127.
17. **Arcos-García, Á., Alvarez-Garcia, J. A., & Soria-Morillo, L. M.** (2018). Evaluation of deep neural networks for traffic sign detection systems. *Neurocomputing*, 316, 332-344.
18. **Yang, Y., Luo, H., Xu, H., & Wu, F.** (2016). Towards real-time traffic sign detection and classification. *IEEE Transactions on Intelligent Transportation Systems*, 17(7), 2022-2031.
19. **Shustanov, A., & Yakimov, P.** (2017). CNN design for real-time traffic sign recognition. *Procedia engineering*, 201, 718-725.
20. **Liang, M., Yuan, M., Hu, X., Li, J., & Liu, H.** (2013). Traffic sign detection and recognition based on pyramidal convolutional networks. *Neural Computing and Applications*, 24(3-4), 613-624.
21. **Kingma, D. P., & Ba, J.** (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
22. **He, K., Zhang, X., Ren, S., & Sun, J.** (2016). Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770-778.
23. **Simonyan, K., & Zisserman, A.** (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
24. **LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P.** (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278-2324.
25. **Ojala, T., Pietikainen, M., & Maenpaa, T.** (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on pattern analysis and machine intelligence*, 24(7), 971-987.
26. **Dalal, N., & Triggs, B.** (2005). Histograms of oriented gradients for human detection. *2005 IEEE computer society conference on computer vision and pattern recognition*, 1, 886-893.
27. **Viola, P., & Jones, M.** (2001). Rapid object detection using a boosted cascade of simple features. *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition*, 1, I-I.
28. **Redmon, J., Divvala, S., Girshick, R., & Farhadi, A.** (2016). You only look once: Unified, real-time object detection. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 779-788.
29. **Ren, S., He, K., Girshick, R., & Sun, J.** (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.
30. **Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C.** (2016). Ssd: Single shot multibox detector. *European conference on computer vision*, 21-37.

Internet manbalar

31. **German Traffic Sign Recognition Benchmark (GTSRB)**. (2023). *http://benchmark.ini.rub.de/*
32. **OpenCV Documentation**. (2023). *https://docs.opencv.org/*
33. **TensorFlow Documentation**. (2023). *https://www.tensorflow.org/api_docs*
34. **PyTorch Documentation**. (2023). *https://pytorch.org/docs/*

35. **Scikit-learn Documentation**. (2023). *https://scikit-learn.org/stable/*