A Robust License Plate Recognition System based on Domain Adaptation

Zhikang Hao, Wendi Li, Zhikang Meng

CS766 Group Project

Outline

1. Background Introduction
2. Dataset Prepare
3. Domain Adaptation
    a. Domain Shift
    b. Domain Adaptation Methods
4. Performance Comparison
5. Implemented System
6. Future Work

Background Introduction

The most critical step in recognition process is character identification.

In general, the environment where a classifier works is diversified and different from the environment of training. For example, a plate recognition classifier may be trained a sunny day, and used in a rainy day, and even a snowy night. It means the training environment is usually different from inference environment. Thus, the strong generalization capability is essential for a good classifier.

Thus, the difference between the training environment and the inference environments harms the performance of the recognition model.

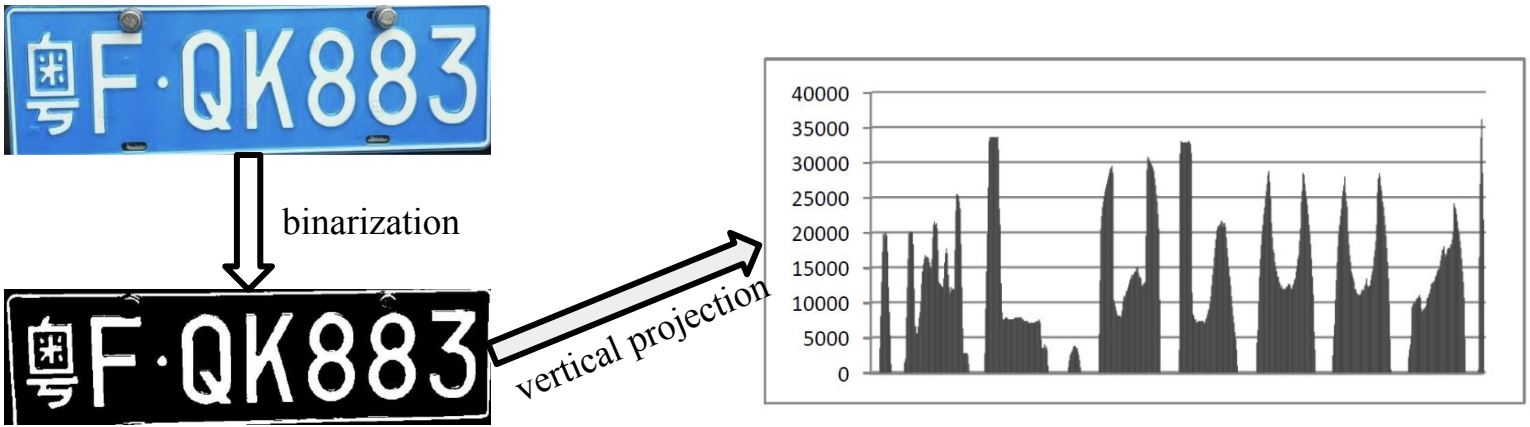
Dataset Prepare

We collect data from the Chinese City Parking Dataset (CCPD), which is a large and comprehensive license plate dataset. The original dataset contains images with labeled bounding box.

Dataset Prepare

To obtain a dataset available for training, we first crop all the license plate parts with the given bounding box, then split them into separate characters.

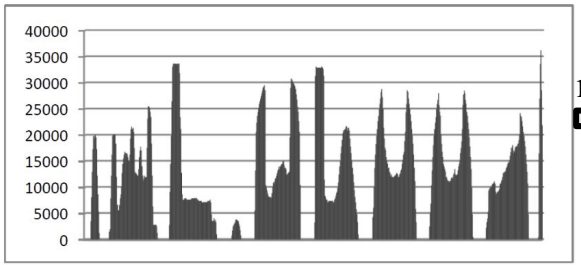
To split each license plate image, we convert the RGB image into binary image, then use vertical projection by column to obtain accumulated histogram.



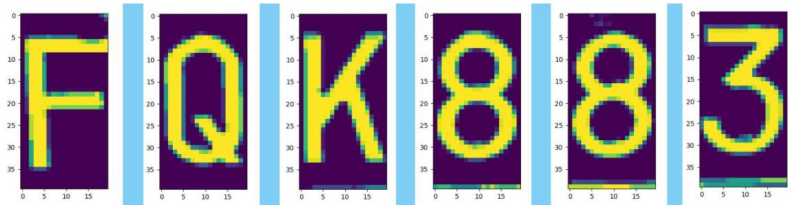
Dataset Prepare

Then, by selecting proper threshold in the vertical projection histogram, we get single character images.

However, there are outliers failed to be correctly separated, so we manually check and fix the dataset. Our final dataset contains 15k images in structured file folders.



set threshold &
manual correction



Domain Shift

This is what our dataset looks like. We can see that the difference is not random. From the images that in our dataset, we can see that the difference is more like a general “shift” among different environments, which means the feature spaces in different environments are almost the same.

Normal

Dark night

Rainy, snow, fog

Far or near to the camera

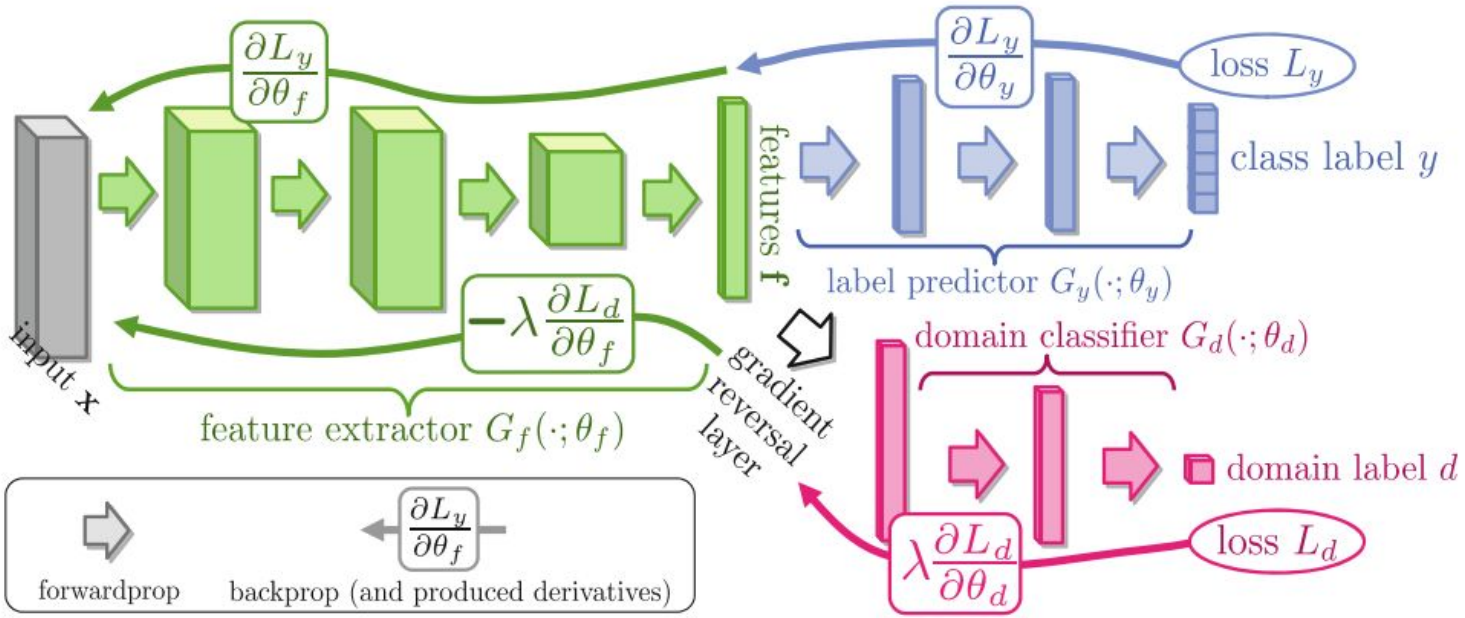
Challenging

CCPD Domain Adaptation Mission

Type	Description
CCPD-Base	The only common feature of these photos is the inclusion of a license plate.
CCPD-DB	Illuminations on the LP area are dark, uneven or extremely bright.
CCPD-Weather	Images taken on a rainy, snow, or fog day.
CCPD-Fn	The license plate is relatively far or near to the camera
CCPD-Challenge	The most challenging images

Tasks	Source domain	Target domain
Domain adaptation 1	CCPD-Base	CCPD-DB
Domain adaptation 2	CCPD-Base	CCPD-Weather
Domain adaptation 3	CCPD-Base	CCPD-Fn
Domain adaptation 4	CCPD-Base	CCPD-Challenge

Domain-Adversarial Neural Networks (DANN) Overview



DANN includes a feature extractor (green) and a label predictor (blue), which together form a standard feed-forward architecture. Domain Adaptation is achieved by adding a domain classifier (red) connected to the feature extractor via a Gradient Reversal Layer (GRL) that multiplies the gradient by a certain negative constant during the backpropagation-based training.

DANN Dissection

The goal of DANN is to find a new representation of the input features in which source and target data could not be distinguished by any **discriminator** network (domain classifier). This new representation is learned by an **encoder** network (feature extractor) in an adversarial fashion. A **task** network (label predictor) is learned on the encoded space in parallel to the **encoder** and **discriminator** networks.

The three network parameters are optimized according to the following two expressions:

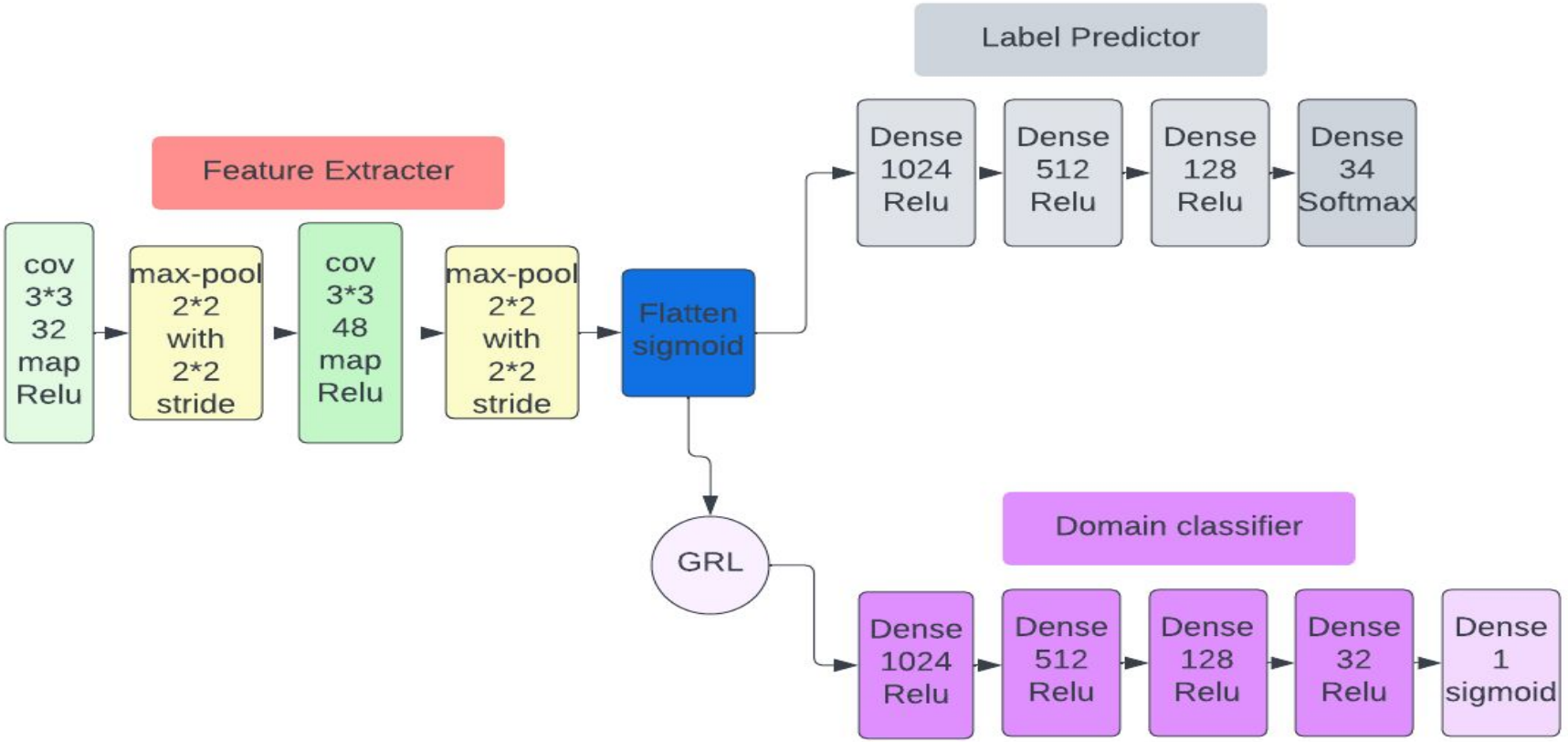
$$\min_{\phi,F} \mathcal{L}_{task}(F(\phi(X_S)), y_S) - \lambda \left(\log(1 - D(\phi(X_S))) + \log(D(\phi(X_T)))\right)$$

$$\max_{D} \log(1 - D(\phi(X_S))) + \log(D(\phi(X_T)))$$

- $(X_S, y_S), (X_T)$ are respectively the labeled source data and the unlabeled target data.
- ϕ, F, D are respectively the **encoder**, the **task** and the **discriminator** networks
- λ is the trade-off parameter.

The adversarial training is done through a **reversal gradient layer** placed between the **encoder** and the **discriminator** networks. This layer inverses the gradient sign in backpropagation, thus the two networks are optimized according to two opposite objective functions.

DANN Architectures: Implementation choice



Domain Adaptation Methods - CCSA

The goal of this algorithm is to make the distance between encoded features of data with same label as close as possible, while distance between different label as far as possible. The encoder is a modified version of MNIST CNN framework.

$$\mathcal{L}_{CCSA}(f) = \mathcal{L}_C(h \circ g) + \mathcal{L}_{SA}(g) + \mathcal{L}_S(g)$$

$$\mathcal{L}_{SA}(g) = \sum_{a=1}^{C} d(p(g(X_a^s)), p(g(X_a^t)))$$

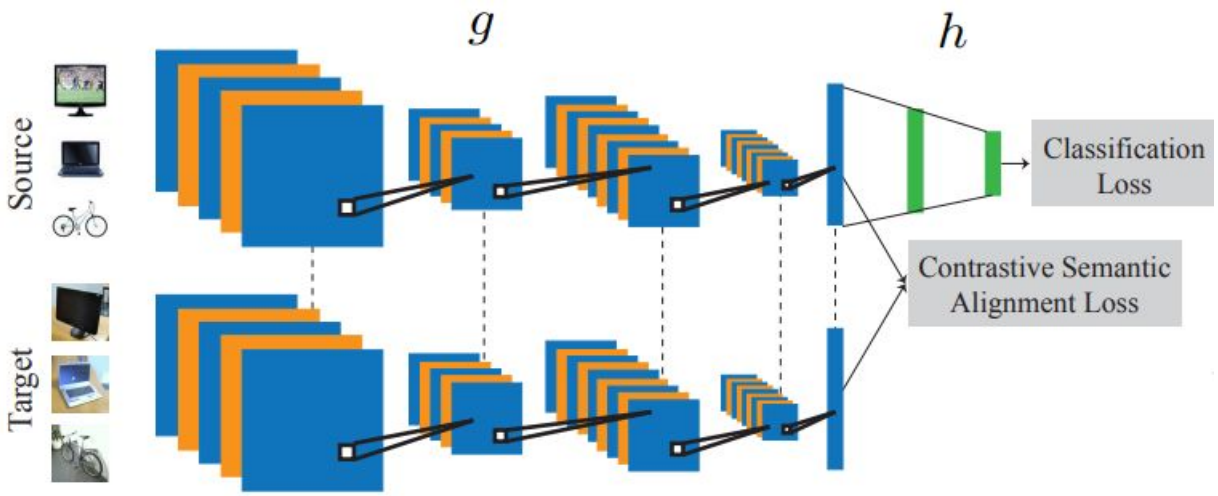
The distance between data within same class

$$\mathcal{L}_S(g) = \sum_{a,b|a\neq b} k(p(g(X_a^s)), p(g(X_b^t)))$$

The distance between data in different class

g is encoding method which extracts the features of input data (images in this case)
h is the task method which classify the input features of source data

Domain Adaptation Methods - CCSA



The classification loss refers to loss on classifier
The CSA loss refers to loss on aligning data distribution with same labels

Domain Adaptation Methods - CCSA

However, obtaining promising classification is challenging when target data on shifted domain is scarce.

As a result, our algorithm compute average pairwise distances between points in the feature space where,

$$d(g(x_i^s), g(x_j^t)) = \frac{1}{2}\|g(x_i^s) - g(x_j^t)\|^2 \quad k(g(x_i^s), g(x_j^t)) = \frac{1}{2}\max(0, m - \|g(x_i^s) - g(x_j^t)\|)^2$$

$$\mathcal{L}_{SA}(g) = \sum_{i,j;\ y_i^s = y_j^t} ||g(x_i^s) - g(x_j^t)||^2 \quad \mathcal{L}_S(g) = \sum_{i,j;\ y_i^s \neq y_j^t} \max(0, m - ||g(x_i^s) - g(x_j^t)||^2)$$

$$\mathcal{L}_{CCSA} = \gamma \mathcal{L}_{task}(h \circ g) + (1 - \gamma)(\mathcal{L}_{SA}(g) + \mathcal{L}_S(g))$$

Gamma is the trade off factor, weighing classification loss and alignment loss

Performance Comparison

We compare the domain adaptation methods to illustrate their effectiveness.

The table below shows their classification performances of different methods.

There are significant improvements when using domain adaptation models.

	Challenge	Db	Fn	Weather
SVM	17.34%	19.90%	22.22%	19.62%
Logistic Reg.	79.92%	75.17%	91.76%	87.64%
DANN	86.86%	88.66%	97.53%	96.25%
CCSA	85.04%	88.82%	98.46%	97.46%

Implemented System

This is what our system looks like. You can try it at https://share.streamlit.io/wendili-cs/cs766_proj/main/st_interface.py