EXPANDING LICENSE PLATE MATCHING CAPABILITIES WITH SECONDARY SELF-LEARNING ALGORITHM

Stephanie R Hargrove, PhD
Department of Civil and Environmental Engineering
The University of Tennessee, Knoxville
311 JD Tickle Building
Knoxville, TN 37996-2313
Phone: (865) 406-3837
Email: shargro1@utk.edu

Hyeonsup Lim
Department of Civil and Environmental Engineering
The University of Tennessee, Knoxville
311 JD Tickle Building
Knoxville, TN 37996-2313
E-mail: hlim4@utk.edu

Lee D. Han, PhD (Corresponding Author)
Department of Civil and Environmental Engineering
The University of Tennessee, Knoxville
319 JD Tickle Building
Knoxville, TN 37996-2313
E-mail: lhan@utk.edu

5,988 words + 6 figures (1,500 words) = 7,488 words
ABSTRACT
To perform the post-processing matching of license plates between two license plate recognition (LPR) stations a self-learning matching algorithm is employed. The key component of this algorithm is an Association Matrix that is a unique “translator” associating two LPR units in terms of how they may see or recognize the same characters differently, for a host of reasons. This Association Matrix consists of primarily high-confidence matches between two LPR stations estimated directly from a set of matched character pairs. The matching algorithm’s performance decreases as the distance between two LPR stations increases. This is due to vehicles no longer travelling within an average travel time window and/or a low sample of vehicles travelling between the two LPR stations. This paper proposes using a third LPR station to generate additional information to derive a better association matrix for an existing pair of LPR stations, thus replacing the existing learned association matrix. In other words, the added LPR unit facilitates a secondary or transferred learning to improve the matching performance of the first two units, even after the third LPR unit is subsequently removed. To evaluate this derived association matrix, we employ two simulations to 1) determine when the newly derived matrix should be used and 2) evaluate the overall performance of license plate matching.
INTRODUCTION

License plate recognition (LPR) technology has been widely applied to numerous transportation applications including automated speed and law enforcement, vehicle tracking, and automated highway tolling. All of these applications require LPR technology to match a license plate at two or more locations. In order to do so without additional post-processing, manual labor particularly, each license plate string (sequence of characters) must be identified correctly to declare a match. If just one character is misread then a match cannot be declared without further efforts and delay, which typically involves human intervention.

License plate recognition technology uses optical character recognition (OCR) engines to identify the text strings of license plates. The matching of OCR recognized license plates is far more complicated than the matching of traditional OCR text, such as text from books. The matching of traditional OCR text has the benefit of readily available dictionaries containing a finite number of vocabulary or strings, context to help determine the likely meaning of the word, and a standard syntax for all characters. On the other hand, license plates strings almost never have meanings and multiple potential syntaxes; one cannot even be certain a plate string was recognized correctly by the LPR algorithm without manual verification. Though, not all is lost, because the process of matching two license plate strings can go beyond looking at the string as a whole and instead use the sequence of comparisons of individual characters.

Consider, for example, that license plate strings $X$ and $Y$ are read at two LPR stations, with the result $X=\text{"ABC123"}$ and $Y=\text{"A8C1Z3"}$. By comparing these two strings, one can make a guess as to whether or not $X$ and $Y$ are a match. To perform the matching of these strings, Oliveira-Neto et al. proposed using the Levenshtein edit distance (ED) technique $[1, 2]$. By applying the Levenshtein ED technique to the example, two fundamental operations (the substitutions “B” to “8” and “2” to “Z”) are required to convert $X$ to $Y$; hence, the total edit distance is two. If the edit distance between the two strings falls below an assigned threshold value, a “match” is declared with some level of certainty.

By examining the predictability of OCR character recognition patterns (1/I, 0/O, 2/Z, 8/B, and 5/S), one could make a more educated guess as to whether $X$ and $Y$ are a match. Oliveira-Neto et al., consequently, proposed a self-learning license plate matching algorithm that included a generalized ED technique with a weight function $[3, 4]$. The generalized ED technique assigns different weights to the edit operations as a function of the character; allowing a measure of similarity between the two plate strings. This technique is reliant on an association matrix to provide the measure of similarity of whether a character, e.g., “B,” recognized by an upstream LPR station is recognized as “B” or “8” or any other characters at the downstream station. These measures of similarity are based on the performance of an associated pair of LPR stations.

LPR technology can achieve different levels of accuracy depending on the hardware and software of the camera, set-up (mobile or stationary), location (side of road or overpass), and on-site calibration. Outside factors also play a role in the level of accuracy (e.g., traffic and weather conditions). The two facets of LPR technology that have the largest effect on the overall performance are the capturing and reading of license plates $[1]$. The capture rate is the rate of successful plate recognitions in the field of view. This is commonly affected by uncontrollable outside factors and parameters pertaining specifically to the camera’s hardware, installation, or on-site calibration. The read rate is the rate of correctly interpreting an entire license plate. This rate is based solely on the performance of the OCR engine. The accuracy of these facets is
commonly uncontrollable by the data output user and dependent on the performance of the LPR technology.

From experience, the typical read rate of LPR cameras rarely surpass 80%, and more commonly perform at 60% or less [1, 3-5]. Consequently, the portion of license plates strings that are correctly recognized and matched between a pair of LPR stations drop down to 35% or less. The current license plate matching algorithm has been tested and proven acceptable at plate matching for sequential LPR stations with reasonable distance between stations (a few miles). However, there are challenging cases where LPR stations are non-sequential, spaced far apart (over 100 miles), experience minimal matches between them, and/or have poor OCR read rates. For these cases, it can be difficult to establish strong association matrices that yield satisfactory matching results.

In order to establish an association matrix that could yield satisfactory matching results for the aforementioned challenging cases, more information is needed. We propose, by employing a third, and perhaps temporary, LPR station, additional information can be gathered to create a secondary self-learning algorithm and derive a replacement association matrix for the initial pair of LPR stations. To evaluate this secondary algorithm, we employ two simulations to 1) determine when and if the derived association matrix should be used and 2) evaluate the overall performance of license plate matching for both algorithms.

REVIEW OF ASSOCIATION MATRIX

The association matrix is the foundation of the license plate matching algorithm. If a poor association matrix is used, the result is low matching rates and/or increased false matches [4]. The following section defines the association matrix and all current estimation methods.

Definition

The association matrix \( A \) between two LPR station \((g, h)\) is a \( N \) by \( N \) square matrix whose elements are the conditional probabilities \( p(b|a) \) of observing a character reading \( b \), \( b \in N \), in station \( h \) for a given character reading \( a \), \( a \in N \), in station \( g \) [3]. The set \( N \) is the set of possible alpha-numeric characters \( \{0,1,2,\ldots,9,A,B,\ldots,Z,\lambda\} \), where \( \lambda \) is the null character. The null character can be more than just an unknown symbol; it can also represent deletions and insertions in the plate string text. When observing the association matrix, the probability of correctly matching a character is seen in the main diagonal elements and the misreading of a character in the off-diagonal elements. Each row of \( A \) refers to a given character recognition at station \( g \), with the column referring to the associated reading at station \( h \).

An association matrix’s elements can be represented in two forms: unitary (counts of matches) and statistical (conditional probability) values [3, 4]. Figure 1 displays an example of a unitary association matrix- where each value represents the number of matches between each character.

To estimate the \( p(b|a) \) of matrix \( A \), the following equation shows the relationship of the unitary and conditional probability values:

\[
p(b|a) = \frac{\rho_{ab}}{\rho_a}
\]  

(1)
where $\rho_{ab}$ is the frequency character $b$, $b \in N$ is associated to character $a$, $a \in N$ in the unitary association matrix and $\rho_a$ is the number of times character $a$ has been matched at LPR station $g$.

| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**FIGURE 1** Example of unitary association matrix.

There are three types of association matrices discussed in this paper; the definition of each is as follows:

1. **Ideal Association Matrix $A_I$**—is an association matrix, which associates perfectly the conditional probability of matches between two LPR stations. This is the ideal matrix hoped to be achieved during the learning process of the matching algorithm.

2. **Learned Association Matrix $A$**—is an association matrix, which is learned directly from comparing output strings from a pair of LPR stations used for matching. This is self-learned using the original matching algorithm.

3. **Derived Association Matrix $A^*$**—is an association matrix created by multiplying two association matrices connected with a shared LPR station. This is created using two learned association matrices for the secondary self-learning algorithm.

**Example of an Ideal Association Matrix**

Suppose there are three LPR stations (1, 2, 3) and the character set used for the license plates has only two elements, either “A” or “B”; thus, the association matrix is a 2 by 2 matrix. The two characters have the same probability of being captured (50% for each); also, representing the
distribution of the characters. Assume the character read rate of each LPR camera’s OCR engine 
is 80%, for all stations. Then the distribution matrix $D$ and the truth matrix $T$ for the stations will 
be as follows:

$$D = \begin{bmatrix} 0.5 & 0.5 \\ 0.2 & 0.8 \end{bmatrix} \quad T = \begin{bmatrix} 0.8 & 0.2 \\ 0.2 & 0.8 \end{bmatrix}, \quad \text{for all stations}$$

Knowing the distribution and truth matrix, the frequency of each match can be calculated. As 
seen in Figure 2, each character has the probability of being captured 50 percent of the time (as 
seen in the first column). By applying the truth matrix, the character will be matched correctly 80 
percent of the time (as seen in the middle two columns). The result represents the frequency of 
each match.

Using Equation 1, the probability of each match can be calculated using the above results. This is 
shown below:

$$p(A|A) = \frac{\rho(A \land A)}{\rho(A)} \quad p(B|A) = \frac{\rho(A \land B)}{\rho(A)}$$

$$p(A|B) = \frac{\rho(B \land A)}{\rho(B)} \quad p(B|B) = \frac{\rho(B \land B)}{\rho(B)}$$

$$\begin{bmatrix} 0.32 + 0.02 \\ 0.32 + 0.08 + 0.02 + 0.08 \\ 0.08 + 0.08 \end{bmatrix} = 0.68 \quad \begin{bmatrix} 0.08 + 0.08 \\ 0.32 + 0.08 + 0.02 + 0.08 \\ 0.08 + 0.02 + 0.08 + 0.32 \end{bmatrix} = 0.68$$

Thus, the ideal association matrix for any station pair $(1 \rightarrow 2, 2 \rightarrow 3, 1 \rightarrow 3)$ will be,
Estimation from Truth Matrices

The truth matrix $\mathbf{T}$, or confusion matrix, is a $N \times N$ square matrix whose elements are the probabilities $p(t|a)$ of each character’s interpretation; where the matrix’s rows represent the character reading $a$, $a \in N$ and the columns representing the true character reading $t$, $t \in N$. Oliveira-Neto et al. [3, 6] proposed estimating the association matrix $\mathbf{A}$ using the following Bayesian expression:

$$\mathbf{p}(b|a) = \sum_{t} \mathbf{p}(b|t) \cdot \mathbf{p}(t|a), \quad t \in N$$

In terms of $\mathbf{A}$ and $\mathbf{T}$, Equation 2 can be written as:

$$\mathbf{A} = \mathbf{T}_g \cdot \mathbf{T}_h^T$$

where $\mathbf{T}_g$ and $\mathbf{T}_h$ are the truth matrices for stations $g$ and $h$, respectively. To make the matrix multiplication possible in Equation 3, $\mathbf{T}_h$ must be transposed; therefore, the rows of $\mathbf{T}_g$ are refer to the character readings at station $g$ and the columns of $\mathbf{T}_h^T$ represent the character readings at station $h$. This process of estimating association matrices from truth matrices, though greatly improving the performance of license plate matching, still has many downsides:

- To acquire the needed ground truth of each character, it requires extensive man hours to extract and visually inspect each license plate; resulting in a very costly and time consuming process.
- In order to estimate a good association matrix, large sample sizes are required. Because of the cost and time to manually identify license plates, this would be impossible for large systems of LPR station (100+).
- The truth matrixes contain separated sets of readings; thus, observing unique plate patterns for each station. The ideal association matrix only contains matches between the associated stations, not the full set of reading for each station. These additional readings could result in the estimated association matrix being skewed from the ideal association matrix.

Estimation from Matching Algorithm

Oliveira-Neto et al. proposed a more efficient way of estimating association matrices by means of a heuristic text-mining algorithm to match plates between two LPR stations [3, 4]. The matching process begins with an initial association matrix, i.e., an identity matrix, to generate a set of learned matches $\mathbf{M}$ using the weighted edit distance technique, as described in [4]. By incorporating the individual character matches from the matched set, an estimated association matrix can be determined.

As stated in [4], the weight function is calculated using conditional probability elements of matrix $\mathbf{A}$, as seen in the following equation:
\[
\gamma(a \rightarrow b) = \log\left(\frac{1}{p(b|a)}\right).
\]

where \(\gamma\) is the weight function relating the corresponding pair of characters \((a, b)\) such that \(\gamma(a \rightarrow b)\) is a non-negative real number assigned to each character’s edit operation. These edit operations are then summed to determine their respective plate string’s edit distance.

Oliveira-Neto et al. shows the estimated association matrix approaches the ideal association matrix as the iterations of the self-learning process progresses [4]. The learning process continues by repeatedly applying the matching process to progressively find more matches and better association matrices. Each iteration \(k\) of the learning process estimates a better matrix \(A_k\) directly from a matching set \(M_k\), obtained from the previous matrix \(A_{k-1}\). The matching algorithm will stop learning when the difference between two successive estimations falls below a pre-assigned threshold.

The process of estimating the association matrix is important to attaining the highest level of matching performance and quickest learning speed. The matching algorithm takes a heuristic approach in determining the elements within a learned association matrix. Meaning, the estimated learned association matrix will never be the ideal association matrix and the purpose of the algorithm is to find a matrix sufficient for the immediate goals.

**DERIVING AN ASSOCIATION MATRIX**

When the read rates of LPR cameras are poor and the sample of vehicles is minimal, the original self-learning matching algorithm simply doesn’t have enough data to learn from to produce a high-performing association matrix. This can regularly be seen in LPR stations spaced a great distance apart, e.g., 100 miles, with minimal matching vehicles traversing between the two stations within a reasonable amount of time. In order to improve the association matrix, more license plate matches are needed to observe the performance of the associated LPR stations. The proposed solution is the addition of a third LPR station, either existing or temporarily deployed, to capture additional license plates and to implement a secondary self-learning algorithm. Theoretically, the third LPR station could be located in any synchronization with the other two LPR stations, just as long as there are shared captured matches between all stations. The additional license plate matches captured at the third LPR station will then be used to create two learned association matrices that are used to derive a more accurate association matrix for the initial pair of LPR stations.

The newly derived association matrix will contain the same elements as mentioned in Equation 1; the only difference is how it is estimated. Similar to the estimation process using truth matrices, the derived association matrix is estimated using matrix multiplication of the association matrix between two pairs of LPR stations sharing a common LPR station. For example, if two existing LPR stations 1 and 2 where supplemented with a third LPR station 3, then two additional association matrices \(A_{13}\) and \(A_{32}\) are estimated from the creation of two additional station pairs \((1 - 3\) and \(3 - 2\)). When \(A_{13}\) and \(A_{32}\) are multiplied the result is the derived association matrix \(A_{12}^*\) between LPR stations 1 and 2. In terms of the matrices the equation is represented as follows:
Proof

Suppose three stations are capturing license plates and the truth matrices \((T_1, T_2, T_3)\) for all stations are known. With the distribution matrix \(D\) of true characters, the distribution of read characters at each station can be calculated by multiplying the true matrices of each station as follows:

\[
D \times T_1 = R_1, \quad D \times T_2 = R_2, \quad \text{and} \quad D \times T_3 = R_3
\]

where \(R\) is a \(37 \times 1\) matrix with elements containing the number of reads for each character in set \(N\). In the set of matches from station \(i\) to \(j\), the association matrix \(A_{ij}\) contains the conditional probability to get each character in station \(j\) given the character in station \(i\). Thus, the set of read characters in station \(j\), \(R_{ij}\), can be simply obtained by multiplying the set of read characters in station \(i\), \(R_{i}\), and the association matrix \(A_{ij}\).

\[
\begin{align*}
R_{12} \times A_{12} &= R_{22}, \quad \text{where } R_{12} \subset R_1 \text{ and } R_{22} \subset R_2 \\
R_{23} \times A_{23} &= R_{33}, \quad \text{where } R_{23} \subset R_2 \text{ and } R_{33} \subset R_3 \\
R_{13} \times A_{13} &= R_{33}, \quad \text{where } R_{13} \subset R_1 \text{ and } R_{33} \subset R_3
\end{align*}
\]

To get the derived association matrix, we need to have the same set of matches for the different pairs of stations (the associated true characters are the same), which are the sub-set of all read characters in each station.

If \(R_{12} = R_{13}, R_{22} = R_{23}, R_{33} = R_{33}\) by assuming they are the same set of matches, then:

\[
R_{23} \times A_{23} = R_{12} \times A_{12} \times A_{23} = R_{13} \times A_{13} \times R_{33}
\]

If we assume to have a perfect matching algorithm, the set of read license plates for all stations will be associated perfectly to the distribution of true license plates passing the stations with respect to the true matrices. Then, the calculation can be even more simplified:

\[
R_1 \times A_{12} \times A_{23} = R_2 \times A_{23} = R_1 \times A_{13} = R_3
\]

Therefore,

\[
A_{12} \times A_{23} = A_{13}^* 
\]

Likewise,

\[
A_{13} \times A_{32} = A_{12}^* 
\]

\[
A_{21} \times A_{13} = A_{23}^* 
\]

Theoretical Examination of Deriving an Association Matrix

From the example of the ideal association matrix, we can obtain the probability a single match is added to each cell of the association matrix, which we define here as a frequency matrix. For
instance, there are two cases ‘A-A-A’ and ‘B-A-A’ of the cell ‘A’ to ‘A’ in the example. Therefore, the frequency of the cell ‘A’ to ‘A’ is 0.32 + 0.02 = 0.34. By adding one to the cell each time to a $N \times N$ zero matrix, we can obtain a new association matrix with a frequency of the event and calculate the closeness between the ideal association matrix and the new association matrix. Closeness is the measure of how close two matrices are to each other. Then, the expected closeness for the sample size $n$ (the number of matched characters), will be the sum of the closeness for all possible sets and their frequencies.

If we consider adding a single character match to each cell of an association matrix as an event, then there are $N \times N$ number of events at each time a single character match is added. The number of all possible events for building the learned association matrix is $(N \times N)_1 \times (N \times N)_2 \times \ldots \times (N \times N)_n = n^{(N^2)}$. Since the derived association matrix is created by multiplying two association matrices, the number of all possible events for building the derived AM is $n^{(N^4)}$. Due to the exponential growth of the computational work load, we examined the expected closeness to the ideal association matrix for both the learned and derived association matrices, up to the sample size of 5. (For $n = 5$ and $N = 2$, the number of all possible events for building the derived AM is 152,587,890,625.)

With this sample exercise, the expected closeness for each sample size ($n$) was calculated. Figure 3 shows how close the learned and derived association matrices are to the ideal association matrix, for the sample size of 1 to 5 ($1 \leq n \leq 5$) and the LPR camera accuracies of 70% to 90%. The calculation was assumed using a 2 by 2 matrix ($N = 2$), however, the results could change with a different size of matrix.

The result implies the derived association matrix is closer to the ideal association matrix for low LPR camera accuracies and small sample sizes (number of matched characters). Note that the learned association matrix is closer to the ideal association matrix when accuracy is 90%; however, this does not guarantee higher performance of LPR matching algorithm with the learned association matrix since there are many factors affecting the performance of LPR matching.
When there are many sources of variance affecting the performance of a LPR camera, it is necessary to have a large sample size to achieve a desired error precision. More specifically, it is important to know the needed amount of data to obtain an accurate estimate of the learned association matrix. Oliveira's initial self-learning algorithm required approximately 60,000 characters to estimate a learned association matrix \[6\]. This approximation may no longer hold true, since advancements can be made with a secondary self-learning algorithm aiming for a drastic reduction in sample size. But by simulating a license plate OCR dataset, the ability to examine a limitless number of characters is gained. Monte Carlo simulation, therefore, becomes a strong tool for examining the relationship between LPR camera accuracy and required sample size. From this relationship we can 1) determine when the derived association matrix should be used and 2) evaluate the overall performance of license plate matching with the derived and learned association matrix.

**When Should We Use the Derived Association Matrix?**
The derived association matrix is theorized to be the solution to the shortcoming of the learned association matrix caused by poor accuracy and small sample sizes. Based on the proof and theoretical estimation, a procedure was designed to determine when the derived association matrix should be chosen over the learned. This simulation is based on an idealistic situation where all characters are matched and the need for a travel time window is removed. The following discuss the steps of the simulation, along with an example and the experimental conditions. The set of characters, matches and association matrices are simulated using the following:

---

**FIGURE 3 Theoretical closeness of derived association matrix.**
1. Generate a distribution matrix \( \mathbf{D} \) to represent the distribution of all true characters. The matrix’s elements contain the cumulative conditional probability for each character within \( N \). Matrix \( \mathbf{D} \) will be the same for all stations (there is no missing/inserted characters between the stations).

2. Generate a truth matrix \( \mathbf{T} \) for each LPR station, which represents the character matching accuracy of the LPR cameras. The elements of these matrices contain the conditional probability across all columns.

3. Generate a random number \( x \), where \( 0 \leq x \leq 1 \), to determine the \( i \)-th true character value for all station with respect to the cumulative conditional probability in matrix \( \mathbf{D} \). This is only done once for each character.

4. For the \( i \)-th true character which was generated in step 3, assign the character read by the OCR at each station by matching the random numbers to the respective cumulative conditional probability of matrix \( \mathbf{T} \) for each LPR station. This is repeated for each true character captured at all LPR stations.

5. Based on the predetermined sample size, the simulation creates a sample of read characters from step 4. Then, every time when a character is created, a learned association matrix \( \mathbf{A}_{12} \) is updated by adding the pair of characters read at station 1 and 2, and so forth for other pairs of stations.

6. Once the learned association matrices are calculated each time in the step 5, a derived association matrix \( \mathbf{A}_{13}^{*} \) is calculated by multiplying \( \mathbf{A}_{12} \) and \( \mathbf{A}_{23} \).

7. Then, the closeness to the ideal association matrix from both the learned association matrix \( \mathbf{A}_{13}^{*} \) and \( \mathbf{A}_{13}^{*} \) are calculated by Equation 6.

8. To capture the impact of accuracy of the LPR cameras, the process is repeated from step 2 and step 7 with different accuracy of the LPR cameras. For example, there are two stations that only capture characters “A” and “B,” meaning the association matrix is a 2 by 2 matrix. Characters “A” and “B” have a license plates distribution of 40% and 60%, respectively. This distribution is show in matrix \( \mathbf{D} \) below:

\[
\mathbf{D} = \begin{bmatrix} "A" & "B" \\ 0.4 & 0.6 \end{bmatrix}, \quad \mathbf{D}^c = \begin{bmatrix} 0.4 & 1 \end{bmatrix}
\]

where \( \mathbf{D}^c \) is the cumulative distribution. LPR stations 1 and 2 both have read rates very close to 68%. The cumulative conditional probability of truth matrices \( \mathbf{T}_1^c \) and \( \mathbf{T}_2^c \) are calculated based on the read rates and are as follows:

\[
\mathbf{T}_1^c = \begin{bmatrix} 0.7 \ 1 \\ 0.35 \ 1 \end{bmatrix} \quad \text{and} \quad \mathbf{T}_2^c = \begin{bmatrix} 0.66 \ 1 \\ 0.3 \ 1 \end{bmatrix}.
\]

For a sample size of four characters, four random numbers (.88, .73, .51, .21) are generated and compared to matrix \( \mathbf{D}^c \) to determine the true character values (B, B, B, A). To determine the captured character at LPR station 1, another set of random numbers (.37, .61, .18, .33) is generated and compared to \( \mathbf{T}_1^c \) with the results (B, B, A). The last step is repeated for the second LPR station using \( \mathbf{T}_2^c \) with the result of (B, B, B, A). For this sample, LPR station 1 misread the third character while station 2 read all characters correctly. From these results a learned association matrix \( \mathbf{A}_{12} \) is calculated. The following is matrix \( \mathbf{A}_{12} \) shown in count and probability form:

\[
\mathbf{A}_{12} = \begin{bmatrix} 1 & 1 \\ 0 & 2 \end{bmatrix} = \begin{bmatrix} 0.5 & 0.5 \\ 0 & 1 \end{bmatrix}
\]
The following experimental conditions were used to evaluate the closeness of the association matrices:

- The use of travel time is exempted from the matching process. Though travel time is an important procedure in the matching algorithm, it remains important to measure the closeness of the self-learning matching without the parameter of a travel time window.
- When determining read rate of LPR stations, the simulation assumes all LPR cameras perform at the same level of accuracy and generates a truth matrix based on that accuracy. Identical accuracy is not common in the real world, but setting a constant accuracy for each simulation aides in a better understanding of the relationship between LPR read rates and sample sizes.
- The simulation captures one character at a time, versus capturing a whole license plate string. Meaning the simulation does not consider how many characters are in each plate, since an association matrix is updated based on character-basis. In doing this, the reliance on the position or sequence of a character during recognition is removed.
- When the characters are captured at each station, the simulation assumes all are matched perfectly, although they may or may not be read correctly. This allows us to see the pure impact of accuracy of the LPR cameras on the closeness to the ideal association matrix from the learned and derived association matrix, even though the accuracy affects the performance of the matching algorithm.
- The accuracy of the LPR camera is the probability of reading a character as its original form (e.g., ‘a-a’, ‘b-b’)
- The sample size number is the number of characters matched by the algorithm (not necessarily saying the number of possible matches unless the capture rate of the algorithm is 100%)

Closeness of Association Matrices

To measure the closeness of the derived and learned association matrices, they are compared to the ideal association matrix used from the simulation. The following equation calculates the closeness (difference) in the two association matrices:

$$\text{closeness} = \frac{\sum_{i=1}^{N} \sum_{j=1}^{N} |a_{ij} - b_{ij}|}{\sum_{i=1}^{N} \sum_{j=1}^{N} |b_{ij}|},$$

where

- $b_{ij}$ is the elements with row $i$ and column $j$ of the ideal AM,
- $a_{ij}$ is the elements with row $i$ and column $j$ of the compared AM.

What is the Performance of the Derived Association Matrix?

Although the first experiment examines the closeness between the derived and learned to the ideal association matrix, the closeness is not enough to see the performance of matching, such as matching rate. The first experiment assumed the algorithm could match every possible pair perfectly to purely see the impact of accuracy of OCR in determining whether the derived or
learned association matrix should be used. However, the performance of matching algorithm
decreases as the accuracy of OCR decreases.

As a result of a low performing matching algorithm, the association matrices of multiple pairs of
stations will likely have different sets of matches. Since the derived association matrix works
well with the same set of matches (as discussed in the proof), the low accuracy of OCR will
result in the derived association matrix worsening.

To evaluate the performance of the derived and learned association matrix for varying LPR
accuracy, the matching algorithm and travel times were added to the simulation to perform a
real-world assessment. The sets of LPR stations, matches and association matrices are simulated
using the following:

1. Generate a distribution matrix $\mathbf{D}$ to represent the distribution of all true characters. The
   matrix’s elements contain the cumulative conditional probability for each character
   within $N$. Matrix $\mathbf{D}$ will be the same for all stations (there is no missing/inserted
   characters between the stations).
2. Generate a truth matrix $\mathbf{T}$ for each LPR station, which represents the character matching
   accuracy of the LPR cameras. The elements of these matrices contain the conditional
   probability across all columns.
3. Generate $n$ vehicles with the travel times at each station with the normal distribution with
   a mean of 60 and a standard deviation of 10. The time window was set according to the
   normal distribution with the $z$ value of 1.96 to capture approximately 95% of true
   matches from the population. Based on the matrix $\mathbf{D}$ in step 1, generate 6 true characters
   for each vehicle.
4. Based on the true matrix $\mathbf{T}$ in step 2, generate read license plates for every vehicles at all
   stations. Some of the vehicles may not be captured depending on the capture rate of
   license plates.
5. Run the original self-learning algorithm to get the learned association matrices of each
   pairs of stations.
6. Calculate the derived association by multiplying the two sets of connected learned
   association matrices, which were obtained in step 5.
7. Run the matching algorithm without learning to evaluate the performance of the matching
   using the learned association matrices.
8. Run the matching algorithm without learning to evaluate the performance of the matching
   using the derived association matrices.

The following experimental conditions were used to evaluate the performance of the association
matrices.

- In contrast to the first experiment, the travel time of vehicles were also considered in this
  simulation to reflect more realistic license plate matching. The narrow travel time
  window in the LPR matching algorithm makes the algorithm faster, but the matching rate
  lower unless the travel time window is large enough to capture all true matches.
- Likewise the first experiment, the simulation assumes all 3 LPR cameras perform at the
  same level of accuracy and generates a truth matrix based on that accuracy. However, as
  this simulation considers license plates with 6 characters, the accuracy includes not only
the accuracy of OCR, but also the capture rates of license plates and characters in each plate.

Performance of LPR matching

Using the set of matches from steps 8 and 9 of the performance simulation, the positive matching rate ($pmr$), false matching rate ($fmr$) and matching rate ($mrate$) are calculated based on the matching results, as seen in the following equations:

\[
pmr = \frac{|M| - |F|}{|M'|},
\]

\[
fmr = \frac{|F|}{|M|},
\]

\[
mrate = \frac{2}{\left(\frac{1}{1 - fmr}\right) + \left(\frac{1}{pmr}\right)},
\]

where $F = \{(X_m, X_n) : (X_m, X_n) \in M, X_m' \neq X_n' \}$ is the set of false matches from $M$, $M' = \{(X_m, X_n) : X_m' = X_n' \}$ is the total set of true matches, with $M$ as defined before [4]. The $mrate$ is the harmonic mean, a common measure of rates, of the fraction of matches that are the actual matches ($1 - fmr$) and the fraction of positive matches that are identified ($pmr$).

RESULTS

When Should We Use the Derived Association Matrix?

Figure 4 shows the closeness to the ideal association matrix and the number of characters, i.e., sample size, needed for the learned and derived association matrices. Each image represents a different LPR character read rate ranging from 40 to 70%. The solid line represents the closeness of the derived association matrix to the ideal association matrix, whereas the dotted line represents that of the learned association matrix.

As seen in Figure 4, the closeness value to the ideal association matrix gets smaller as the sample size increases for both the learned and derived association matrices (i.e., they become more alike). Also, the derived association matrix is closer to the ideal association matrix up to a certain number of matched characters for the each accuracy of LPR cameras.

The intersecting point, where the derived association matrix no longer is closer to the ideal association matrix, gets smaller (moving to the left in the Figure 4) as the accuracy increases.

This is summarized in Figure 5, and allows for examination of when the derived association matrix should be used based on the accuracy of the LPR cameras. Each point of Figure 5 represents an intersecting point where the learned association matrix becomes closer to the ideal association matrix.

Note that this simulated experiment assumed the matching algorithm is perfect; therefore, the characters have 100% positive matching even though the read characters may be incorrect. This
is dependent on the accuracy of LPR cameras and the closeness to the ideal association matrix may not reflect the performance of matching algorithm thoroughly.

**FIGURE 4** Differences between association matrices with shared read rate.

**FIGURE 5** Allowable number of sample size and OCR character accuracy to use derived association matrix.
What is the Performance of the Derived Association Matrix?
To measure the performance of matching algorithm depending on whether the learned or derived association matrix is used, an additional simulated experiment was performed. In this simulation, as described, the Oliveira-Neto et al.’s self-learning LPR algorithm was used with the consideration of travel time [4].

Figure 6 shows the matching rate performance curves by the number of matched plates for the learned and derived association matrices. The results are averaged from three different simulations with the same accuracy of LPR cameras. In the results, the matching algorithm using the derived association matrix performed better most of time compared to the learned association matrix. Specially, the gap of performance is large when the number of matched license plates is small.

As shown in Figure 6, to achieve a 90% matching rate, the matching algorithm using the derived association matrix only requires about 300 matched license plates while the learned association matrix requires more than 1,200 matches. To achieve the 90% positive matching rate, the derived association matrix requires only about 500 matches whereas the learned association matrix had to learn more than 1,800 matches.

Summary
To evaluate the derived association matrix, we employ two experiments to 1) determine when the matrix should be used and 2) evaluate the overall performance of license plate matching. The first experiment examined the closeness of the derived and learned association matrix to the ideal association matrix. The second simulated a case study of license plate matching that evaluated the performance of the derived and learned association matrix. In order to examine multiple parameters, the two experiments where compared with different conditions. Experiment 1 used characters such as the sample, no travel time constraints, and 100% matching. Experiment 2 used license plates as the sample, had a travel time constraint, and the matching was based on the LPR accuracy; consequently, resulting in a perfect world versus real world analysis.

The first experiment shows the derived is close to the ideal when the sample size is small and the accuracy is low, but the second experiment shows the performance of matching is still higher by using the derived results even when the accuracy is high. The results show that even with the learned association matrix having the smaller closeness value, that it does not a higher performance rate. It could be that the derived association matrix’s performance does not depend on the closeness to the ideal, but rather the calculation of edit distance or the travel time window in the matching algorithm. Or that having a high accuracy may aid in equal sets of matches for three stations; thus, the derived association matrix outperforms the learned association matrix.
FIGURE 6 $m_{rate}$ performance curve for OCR character accuracy.

CONCLUSION

By utilizing the secondary self-learning algorithm with a derived association matrix, new life can be brought back to LPR stations that may have been forgotten, due to poor location or performance. By providing a sense of where and when the derived association could be implemented, LPR users can now achieve station set-ups that would have failed in the past, due to low number of matches. Not only does this make LPR technology more flexible, but the derived association matrix also showed an increase in learning speed. As shown in the paper, the derived association matrix was able to reach a 90% matching rate with 300 matched license plates while the learned association matrix required 1,200 matches.
With tens of thousands of LPR stations deployed in the United States, it is possible that the derived association matrix is a better alternative for matching license plates over a large network of LPR stations. The derived association matrix could also become a solution for other unsuccessful OCR text matching applications other than license plate text.

Future studies will be required to examine other factors related to the performance of the license plate matching, such as the edit distance criteria and travel time window. The simulation of more than three LPR stations would also be valuable in determining if derived results can be determined from more than two pairs of stations. Most importantly, the formulation of the edit distance needs to be critiqued and possibly even replaced with a new calculation.

ACKNOWLEDGMENTS

The research effort and findings presented herein were sponsored by Tennessee Department of Transportation, USDOT’s Southeastern Transportation Center, and the University of Tennessee’s Chancellor Scholarship program.
REFERENCES


