Integrated Computer-Aided Engineering 21 (2014) 219–234
DOI 10.3233/ICA-140460
IOS Press

Audio-video based character recognition for handwritten mathematical content in classroom videos

Smita Vemulapalli[a] and Monson Hayes[a,b,*]
[a]*Center for Signal and Image Processing, Georgia Institute of Technology, Atlanta, GA, USA*
[b]*Advanced Imaging Science, Multimedia and Film, Chung-Ang University, Seoul, Korea*

Abstract. Recognizing handwritten equations is a challenging problem, and even more so when they are written in a classroom environment. However, since videos of the handwritten text and the accompanying audio refer to the same content, a combination of video and audio based recognition has the potential to significantly improve the recognition accuracy. In this paper, using a combination of video and audio based recognizers, we focus on improving the character recognition accuracy for handwritten mathematical content in videos using audio and propose an end-to-end recognition system. The system includes components for video preprocessing, selecting the characters that may benefit from audio-video based combination, establishing a correspondence between handwritten and the spoken content, and finally combining the recognition results from the audio and video based recognizers. The current implementation of the system makes use of a modified open source text recognizer and a commercially available phonetic word spotter. For evaluation purposes, we use videos recorded in a classroom-like environment and our experiments demonstrate the significant improvements in character recognition accuracy that can be achieved using our techniques.

Keywords: Video preprocessing, handwriting recognition, speech recognition, classifier combination

1. Introduction

Recent years have witnessed a rapid increase in the number of e-learning and advanced learning initiatives that either use classroom videos as the primary medium of instruction or make them available online for reference by the students. As the volume of recorded video content increases, it is clear that in order to efficiently navigate through the videos there is a need for techniques that can help extract, identify and summarize the video content. In this context, and given the fact that the whiteboard continues to be the preferred and effective medium for teaching complex mathematical and scientific concepts [4,60], this paper focuses on how to use both the audio and video to achieve a higher recognition accuracy compared to when only one recognizer is used alone. It is clear that, in any recognition task, one should use all available information to aid in the recognition task. Since instructors who write their lectures on the whiteboard will typically speak what is being written, it is possible to use the audio to improve the recognition accuracy of a character recognizer. The question is how best to combine the outputs of a character and an audio recognizer, and how to use the audio to assist in character recognition. There are many difficult issues that need to be addressed, such as when should audio be used to assist in the recognition? For example, if a handwritten character is correctly recognized by the character recognizer, then any further processing using the audio may result in an error. If the character recognizer is having difficulty in recognizing a character, and there is an ambiguity as to what the correct character is, then how many

*Corresponding author: Monson Hayes, School of Advanced Imaging Science, Multimedia and Film, Chung-Ang University, Seoul, Korea. Tel.: +1 404 462-8257; E-mail: mhh3@gatech.edu.

ISSN 1069-2509/14/$27.50

options should be considered when using the audio to assist in the recognition? When the audio is searched to see if a specific character has been verbalized, over what window should the search be made, and what search terms should be used? The character "2" may be spoken in a variety of ways, depending upon the context in which it is written. It could be spoken, for example, as "two" or "twice" or "square". If two or more utterances for a word are found, which one corresponds to the character that was written? When placing this audio utterance in the context of others, it is important to select the correct one. And how should the context in which a character is spoken be used? In other words, how does one incorporate any knowledge or information about what characters may appear within a neighborhood of an ambiguous character to help resolve the ambiguity? In this paper, we propose a variety of ways to address these and other issues in combining video and audio to recognize handwritten equations.

There is a significant body of research devoted to the recognition of handwritten equations [1,3,6,32,52,63], and to the extraction and recognition of textual content from video [16,17,29,45,61]. While the research presented in this paper is closely related and dependent on advances made in these fields, the focus here is on how to use audio to enhance the performance of a character recognizer. Specifically, this paper presents a recognition system for audio-video based character recognition for handwritten mathematical content in classroom videos.

2. Related work

The research presented in this paper lies at the intersection of four distinct and well-studied specializations: video processing, handwritten text recognition, speech recognition and classifier combination. In the following sections the relevant literature for each of these specializations is reviewed along with some recent work on audio-video content recognition.

2.1. Video preprocessing for text extraction

A method for detecting and tracking text in digital video is proposed in [29], which implements a scale-space feature extractor that feeds an artificial neural processor to detect text blocks. Methods have also been proposed to detect key frames in video for easy retrieval, as well as methods to acquire data about the movement of the tip of the pen or pen strokes on a whiteboard [45,50]. For example, a system that automatically produces a set of key frames representing all the written content on the whiteboard before each erasure is described in [16]. Some of the video content for which it is important to be able to extract textual content from video include recorded lectures for indexing and retrieval in e-learning initiatives [13], video-taped presentations for summarization [24] and commercial and personal videos for searching and cataloging. An advanced technique for extracting text from faded historic documents that is arranged as a complex pattern, with parallels to mathematical content and out of focus handwritten content, is presented in [44].

2.2. Handwritten mathematical text recognition

There is a vast collection of literature related to the recognition of handwritten text, and a comprehensive survey of this research is presented in [38]. Handwriting recognition may be done from a variety of input sources such as paper documents [11,28,43,47], pen-based inputs on an electronic screen [37], video [51] and other specialized input devices [50]. Mathematical content recognition [6,8] presents some challenges that are quite different from those of recognizing text. This is due to the fact that mathematical characters and symbols have different sizes and often they are arranged spatially in a complex two-dimensional structure. For mathematical character recognition, researchers have addressed issues that arise from factors such as the large number of similar symbols (u versus μ or v versus v) that must be recognized, and the lack of a lexicon that may be used for final validation [32].

Research has shown that the recognition accuracy of mathematical expressions can be improved by combining two or more stages [46], and Prusa et al. have shown how to use a two-dimensional grammar to achieve better mathematical formula recognition [40]. Similarly, a hidden Markov model based method that avoids segmentation during pre-processing by making use of simultaneous segmentation and recognition capabilities is presented in [25,26]. Finally, Awal et al. describe an interesting approach for simultaneous optimization of segmentation, recognition and structure analysis that is constrained by a mathematical expression grammar [5].

2.3. Speech recognition

Speech recognition is the process of converting an acoustic signal captured by a microphone or a sim-

ilar device into a sequence of words. Over the last few decades, research in speech recognition has made significant advances that have led to the development of a number of commercial speech recognizers including Dragon Naturally Speaking [10], Microsoft Speech [34] and IBM ViaVoice [58]. There have also been some important research contributions from the academic community with the HTK [19] project from the University of Cambridge, and the Sphinx [49] project from CMU. Nexidia's word spotting tool [35] provides a fast an efficient approach to search for words within an audio stream.

2.4. Classifier combination

The field of classifier combination has been constantly evolving to address the challenges posed by new application domains. A comprehensive survey of classifier combination techniques is presented in [53], which partitions the classifier combination methods along several distinct dimensions, some of which include the way in which the outputs of the classifiers are combined, whether the number of classifiers is fixed or if the combination methods use classifiers from a large pool of classifiers, etc. There is also a large body of research that focuses on generic methods for classifier combination. Lucey et al. [30], for example, have proposed a theoretical framework for independent classifier combination. While some classifier combination methods use another classifier to combine the output of multiple classifiers, others make use of rules and functions to combine the outputs. Some of the combination techniques proposed in our research are adaptations of well known methods such as weighted combination, Borda count [54], and other decision combination techniques [18]. In the context of handwriting and speech recognition, classifier combination techniques have also been used to improve the recognition accuracy of handwriting recognizers [14,41,48,59] as well as speech recognizers [12].

2.5. Audio-video based content recognition

Audio and video signals carry complementary information and often an error in recognizing a spoken character will not be accompanied by an error in recognizing the written character. Therefore, a combination of both information sources can potentially lead to a significant improvement in the recognition accuracy compared to that which is obtained when either one is used alone. Yu et al. [23,62] propose a classifier combination framework for grammar-guided sentence recognition, and present results for spoken email command recognition, where an acoustic classifier and a visual classifier (for recognizing lip movement, and tongue and teeth visibility) are combined. The Speech Pen system with an advanced digital whiteboard recognizes speech and handwriting in the background and provides the instructor with a list of possible next words that allow the instructor to skip manual writing [27].

A comprehensive collection of research in the field of audio-visual speech recognition is presented in [39]. In the context of classroom videos that utilize slides and digital ink, Anderson et al. present an empirical basis for addressing the problem of the automatic generation of full text transcripts for lectures [2]. Their approach relies on matching spoken content with slide content, and recognizing the meaning of the content written by the instructor using digital ink. An investigation of a number of strategies for combining HMM classifiers for improving audio-visual recognition is presented in [31], and based on empirical, theoretical and heuristic evidence, a recommendation is made for using a hybrid of the sum and product rules.

Hunsinger et al. have proposed a multimodal mathematical formula editor that combines speech and handwriting recognition. In [21], they describe the speech understanding module of the system, and in [20] they present a multimodal probabilistic grammar that incorporates the syntactic-semantic attributes of spoken and handwritten mathematical formulas. A system for speech and handwriting data fusion based isolated mathematical symbol recognition is presented in [33]. Although neither the speech nor the handwritten data originates from video, the techniques used to combine the output of the character and speech recognizers are similar to the techniques presented in this paper. However, issues such as ambiguity detection and A/V synchronization are not considered in the aforementioned research. Another relevant research effort, closely tied to [33], relates to the creation of a data set with handwritten and spoken mathematical content [41]. Unfortunately, this data set consists of static image segments containing handwritten content, with the corresponding audio stored in separate files. The absence of a video sequence (for audio and video time-stamping) makes this data set unsuitable for our experiments.

3. System overview

The overall system for recognizing handwritten mathematical text and equations is shown in Fig. 1.

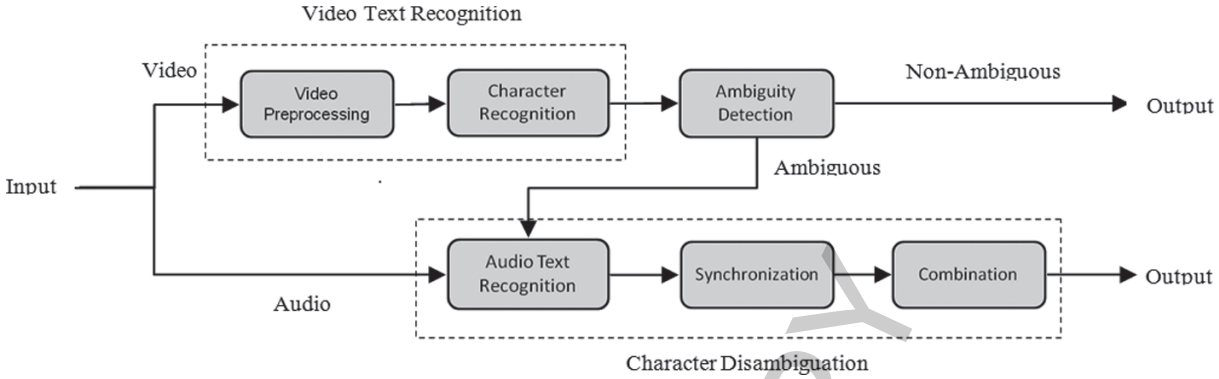


Fig. 1. Components of the audio-video character recognition system.

The first stage is the *video text recognizer* that includes a *video preprocessor* and a *character recognizer*. The video preprocessor includes all of the processing that is required to extract the text that is to be recognized, segment the text into characters, generate timestamps for each character in the video, and tag the location of the characters in the video frame. For each segmented character, the character recognizer then generates a list of one or more possible characters from a dictionary of possible characters. The character recognizer also generates a *score* for each character in the list that represents the recognizer's belief in the correctness of the character name. Since the characters in this list are based only on the video, they will be referred to as *video options.*

Following the video text recognizer is the *ambiguity detector* that determines whether or not the video option with the highest score is likely to be the correct character. If not, then the character to be recognized is classified as *ambiguous*, and two or more video options are selected for further processing to determine which is the correct one. This is done in the character disambiguation stage.

The *character disambiguation stage* consists of three components. The first is an *audio text recognizer* that first assigns one or more audio search terms for each video option, and then searches the audio within some window for the occurrence of these terms. The output of the audio text recognizer is a set of *audio options*, which are occurrences in the audio stream of the video option's character name. Each audio option consists of the audio search term, an audio timestamp, and an audio match score. The second component is the *audio-video synchronizer* that processes the set of audio options and assigns at most one audio option to each video option. The output of this stage is one or more audio/video pairs. The final step is *audio-video combination*. Here, the recognition scores of each audio/video pair are analyzed to produce a final recognized character.

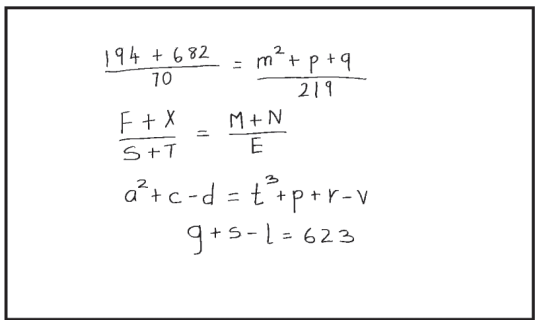
In the following section, we begin by looking at the task of video text recognition, i.e., character recognition that uses only the video.

4. Video text recognition

When capturing the video for video text recognition, a few assumptions are made about the recording process. Although these assumptions are not very restrictive, they simplify many of the preprocessing tasks. First, it is assumed that the entire whiteboard is within the field of view of the camera, and that the whiteboard (the *region of* interest) is easily detected. It is also assumed that the beginning of every recording session has at least one *calibration frame*, which is a video frame with a clean whiteboard without the instructor. In the recording of a lecture, it is assumed that the board is erased completely before the instructor begins a new board, and that the instructor briefly steps away from the board after a complete erasure so that the entire region of interest is unobstructed.

4.1. Video preprocessing

The first step in the video text recognizer is the video preprocessor that performs a number of important tasks [55]. The first is to identify the *region of interest* (the whiteboard), and the second is to detect the frames of interest in the video, which are unobstructed views of the whiteboard just prior to an erasure. Thus, the frame of interest contains all of the characters that are to be detected and recognized. The process of detecting the frame of interest primarily relies on count-



(a)

Image	Time	Coordinates
	32.5	(241, 96, 245,134)
	33.4	(264, 90, 281, 129)
	34.6	(300, 86, 321, 126)
	36.7	(355, 93, 377, 120)
	38.0	(407, 89, 426, 123)
	42.3	(442, 89, 462, 114)
	45.1	(467, 85, 491, 113)

(b)

Fig. 2. Video Text Recognition. (a) Text that has been extracted and segmented into characters from a video frame and (b) the vector $\mathbf{c}(s)$ that consists of the image of the character s, and the time and location that the character appeared on the whiteboard.

ing the number of connected components contained in video frames over the duration of the video, and selecting the frame with the maximum number of connected components.

The next step is *character segmentation*. Assuming that a given frame of interest is free of any occlusions or shadows from the instructor and that individual characters appear as one or more distinct connected components, a connected component analysis algorithm is used to extract the characters [7]. A post-processing step allows for the handling of characters in the dataset that do not appear as a single connected component, such as 'i' and '='. The final step is to produce a video timestamp for each segmented character, which is the time at which the character is written on the whiteboard. An example is given in Fig. 2, which shows a set of segmented characters. Associated with each character is a vector

$$\mathbf{c}(s) = [s,\ t(s),\ \mathbf{l}(s)]$$

that contains the image of the character, s, the time at which the character was written on the board, $t(s)$, and the location of the character on the board, $\mathbf{l}(s)$. Since the location is used only in the structure analysis of an equation [55,56], it will not be used here.

4.2. Character recognition

Once a character has been extracted, it is forwarded to the character recognizer. Although there are many recognizers that could be used, we chose the GNU Optical Character Recognition program or GOCR [15]. It is important to point out that although GOCR is not the best character recognizer for handwritten text, the focus of this paper is not to build a state-of-the-art audio/video character recognizer, but rather to investigate ways in which audio and video may be combined to improve the accuracy of any given character recognition system. If the recognizer were perfect, then there would be no need to use audio to assist in the recognition of characters. However, since no handwritten character recognizer is perfect, then any recognizer that introduces errors or uncertainties in the recognition of characters may be used to study different approaches for audio-assisted character recognition. It should also be pointed out that, in many cases, it may not be possible to use a state-of-the-art recognizer if one is interested in real-time character recognition within a simple platform. Although the techniques and approaches presented in this paper are not tied to any specific recognizer, some of the parameters along with the final recognition rates will be different and depend on what character (and audio) recognizer is used.

Two modifications to GOCR were made. The first was to have GOCR return a set of candidate characters rather than a single recognized character or no match at all. The second was to return a score that is based on the number and the relative significance of the recognition rules that are satisfied. Thus, when the image of a character is passed to the character recognizer, a set of possible characters is generated along with a score that indicates how likely the given character is the correct one. Since these candidate characters are based only on the video, they will be referred to as *video options*, as opposed to *audio options* that are based on the audio as discussed later in Section 6.1. Thus, as illustrated in Fig. 3, the output of the video text recognizer is a set of L video options for each character s, where each video option, $\mathbf{v}_j(s)$, is an ordered pair

$$\mathbf{v}_j(s) = [v_j^c(s), v_j^p(s)]$$

where $v_j^c(s)$ is a character from the dictionary C and $v_j^p(s)$ is the recognition score for that character.

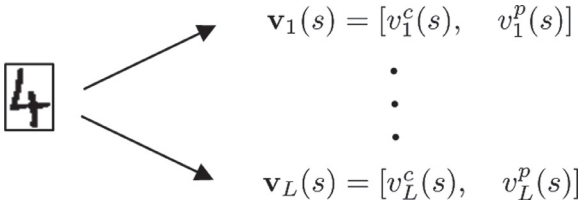


Fig. 3. A character s with L video options.

The current system recognizes the alphabetic characters (upper case and lower case), numbers and basic arithmetic operators. Expanding the dictionary to include other characters, such as Greek letters and more complicated mathematical symbols is straightforward.

5. Ambiguity detection and option selection

After the character recognizer generates a set of video options for a character, the next step (ambiguity detection) is to decide whether or not the option with the highest score is likely to be correct. If it is likely to be correct, then it is output as the final recognized character. However, if it is determined that there is a sufficiently high probability that this option may be incorrect, then two or more options that satisfy some option selection criteria are sent to the audio recognizer to assist in the recognition.

5.1. Character recognition score

The recognition scores produced by the video text recognizer are often not the best metric to use to determine whether or not a recognized character should be classified as ambiguous. Therefore, these scores are mapped to a new set of scores that are better matched to the task of tagging the ambiguous characters. Some commonly used score normalization techniques are discussed in [22], but the approach that is used here is to replace the score with an estimate of the *conditional probability* that the character is correctly classified, given the video match score for that option. More specifically, let G be a function that returns the ground truth for a given character s,

$$G(s) = c$$

The conditional probability is then given by

$$\begin{aligned}&\text{Prob}\left\{v_i^c(s) = G(s) | v_i^p(s)\right\}\\ &= \frac{\text{Prob}\{v_i^c(s) = G(s), v_i^p(s)\}}{\text{Prob}\{v_i^p(s)\}}\end{aligned}$$

Estimating these conditional probabilities is done using a training set along with the ground truth for each character in this set. Thus, the estimate of this conditional probability that will be used as the character recognition score, $\bar{v}_i^p(s)$, is

$$\bar{v}_i^p(s) = \frac{N(v_i^c(s) = G(s), v_i^p(s))}{N(v_i^p(s))}$$

where the term in the numerator is the number of times $v_i^c(s)$ is correctly classified when its score is $v_i^p(s)$, and the term in the denominator is the number of times $v_i^c(s)$ has a score of $v_i^p(s)$. In some cases, such as when there is a limited training set, it may be necessary to divide the range of scores into intervals and estimate the conditional probability given that $v_i^p(s)$ falls within some range of values.

5.2. Character classification

Having an appropriate set of scores for each video option for a given character, it is now necessary to determine whether or not a character should be classified as ambiguous. Those that are ambiguous will be sent to the audio recognizer to assist in the recognition. It may at first seem best to send every character to the audio recognizer for verification or correction, but in those cases where the video option with the highest score is correct, the audio recognizer may find an utterance (or no utterance at all) in the audio thereby making another character more likely and, as a result, introducing an error in the final recognition result. Similarly, if an incorrectly recognized character is not forwarded to the audio text recognizer, then there is no possibility for the error to be corrected. Therefore, it is important to determine which characters have a sufficiently high probability of being incorrectly recognized, and tagging these as *ambiguous* and sending only these to the audio recognizer.

A character is classified as non-ambiguous if its recognition score exceeds some threshold. To perform this classification, two types of thresholds were considered: simple thresholds and character-specific thresholds. In the following, S will be used to denote the set of all characters that are to be recognized, and $D(s)$ will be used to represent the set of all characters in S that are classified as ambiguous. It will be assumed, for simplicity, that the video options for each character, $\mathbf{v}_j(s)$, have been ordered according to their recognition score with the first option having the largest score.

Table 1
Classification of characters in the training set for a given character-specific threshold T

	Top video option	Tag
$N_t(T)$	Correct	Non-ambiguous
$N_f(T)$	Incorrect	Non-ambiguous
$A_t(T)$	Incorrect	Ambiguous
$A_f(T)$	Correct	Ambiguous

5.2.1. Simple thresholds

The first threshold criterion considered for ambiguity detection is one that classifies a character as ambiguous if the option with the largest score is less than some absolute threshold, T_A,

$$D(S) = \{s \in S \mid \bar{v}_1^p(s) < T_A\}$$

The second is to evaluate the ratio of the second largest score, $\bar{v}_2^p(s)$, to the largest score, $\bar{v}_1^p(s)$, and if this ratio exceeds some threshold, T_R, then the character is classified as ambiguous,

$$D(S) = \{s \in S \mid \bar{v}_2^p(s)/\bar{v}_1^p(s) > T_R\}$$

The rationale here is that if the top video option is unequivocally correct, then it should have a score that is significantly larger than the second best video option.

5.2.2. Character-specific thresholds

Since some characters are more difficult to recognize than others, and since a character recognizer will generally have different recognition rates for different characters, having the same threshold for all characters is generally not the best approach to use. Therefore, another approach is to use a different threshold for each character in the dictionary. To set these character-specific thresholds (that will depend on the specific character recognizer that is used), a training set for each character in the dictionary is created. Let $S(c)$ denote the training set for the character c. Each character in $S(c)$ is sent through the character recognizer, and this set is then partitioned into four sets as illustrated in Table 1. This partition, which depends on a threshold T, is generated as follows. Let $N(T)$ denote the set of all characters that would be classified as non-ambiguous using a threshold of T. In other words, the top recognition score for each of these characters is larger than T. As a result, these characters will not be sent to the audio recognizer for further processing, and the video option having the largest recognition score will be the final output. This set is then partitioned into two sets, $N_t(T)$ and $N_f(T)$. The characters in the first set are those for which the video option with the highest score is the correct character, c, and therefore will be correctly recognized. The characters in the second set, on the other hand, are those for which the video option with the highest score is incorrect and will be incorrectly recognized.

All of the characters not in $N(T)$ are in a set denoted by $A(T)$, and these are the characters that would be classified as ambiguous using the threshold T and would be sent to the audio recognizer for additional processing. This set is partitioned into two sets, $A_t(T)$ and $A_f(T)$. The characters in the first set are those that are correctly classified as ambiguous because the correct character is not the one with the highest recognition score. Therefore, further processing may result in the correct recognition of these characters. For those characters in the second set, the video option with the highest recognition score is the correct one. However, since the score does not exceed the threshold T, they are sent to the audio recognizer for further processing and may, eventually, be recognized incorrectly.

For a given threshold, T, the recognition rate for the character c over the training set $S(c)$ is equal to

$$\alpha(T, c) = \frac{|N_t(T)| + \alpha_A^L(|A_t(T) \cup A_f(T)|)}{|S(c)|}$$

where $|A|$ is the number of elements in the set A, and α_A^L is the recognition accuracy for the ambiguous characters when L video options are sent to the audio recognizer. This value is estimated from the training set. The character-specific threshold $T(c)$ is then defined to be the value of T that maximizes the recognition score,

$$T(c) = \arg\max_T(\alpha(T, c))$$

It is important to note that these thresholds depend on the specific character recognizer that is used.

5.3. Option selection

After a character has been classified as ambiguous, it is necessary to determine which set of video options should be forwarded to the audio recognizer to help resolve the ambiguity. If the character recognizer produces N video options for a character, then the naïve approach would be to send all N options to the audio recognizer since this would increase the probability that the correct character would be among the options that are forwarded. However, when the correct character is the one with the highest score, then each additional option that is forwarded would increase the

chances that an error will be made in the final output. On the other hand, if the number of options that are passed is too low, then the chances are higher that the correct character would not be included within this set. Thus, the goal of *option selection* is to choose a set of video options in such a way that the probability of having the correct character within the list is maximized while, at the same time, minimizing the number of options that are in the list.

Three different option selection strategies were considered. If K options are to be forwarded to the audio recognizer, then the first option is simply to select the K video options that have the largest recognition score. Again assuming that the video options have been ordered according to their recognition score, with $\mathbf{v}_1(s)$ having the largest score, the set of video options is

$$O(s) = \{\mathbf{v}_1(s), \mathbf{v}_2(s), \ldots, \mathbf{v}_K(s)\}$$

The second strategy is to select all video options that have a recognition score that exceeds a threshold T. Thus, if $V(s)$ is the set of all video options, then

$$O(s) = \{\mathbf{v}_i(s) \in V(s) \mid \bar{v}_i^p(s) > T\}$$

In this case, the number of video options is variable. The third approach is to select all video options that have a recognition score that exceeds some fraction, T_O of the highest score,

$$O(s) = \{\mathbf{v}_i(s) \in V(s) \mid \bar{v}_i^p(s)/\bar{v}_1^p(s) > T_O\}$$

Here again, the number of options is not fixed. In the event that no video options satisfy the threshold condition, the top one or two options would be selected.

6. Audio-video synchronization

Once the video options for the ambiguous characters have been identified, the audio recognition system is used to determine which of these options, if any, are found in the audio within some interval around the time that the character is written on the board. Since the goal is to *search* for specific phonemes or spoken words, Nexidia's word spotter was used since it is fast, works well for non-standard grammars, and does not require any training [35]. For the case in which two or more audio options are found for a given video option, it is then necessary to perform audio-video synchronization to match the appropriate audio option with the given video option. First, we discuss what is meant by an *audio option.*

6.1. Audio options

When a handwritten character s that is written at time $t(s)$ is classified as ambiguous, an *audio search window* is defined that extends from time $t(s) - t_c$ to time $t(s) + t_c$.[1] Then, for each video option $\mathbf{v}_j(s)$, one or more *audio search terms* are defined for this character $v_j^c(s)$. These audio search terms are the phonemes or words that might be spoken when the character s is written on the board. For example, if $v_j^c(s) =$ "2", then the audio search terms might be "two", "squared", "twice", and "double". The audio is then searched over the given window for each audio search terms, and for each one that is found, an *audio option* is created. These audio options are vectors that contain the audio search term, $a_{j,k}^c(s)$, the time that it occurs in the audio, $a_{j,k}^t(s)$, and an audio match score, $a_{j,k}^p(s)$. Thus, the k^{th} audio option for the j^{th} video option $\mathbf{v}_j(s)$ would have the form

$$\mathbf{a}_{j,k}(s) = [a_{j,k}^c(s),\ a_{j,k}^t(s),\ a_{j,k}^p(s)]$$

An example is shown in Fig. 4 where the character s has two video options. For the first option, $\mathbf{v}_1(s)$, only one audio search term is defined, which is "four", and over the audio search window two audio options are found. However, for the second video option, $\mathbf{v}_2(s)$, which also has only one audio search term, only one audio option is found. Any audio option that has an audio match score that is below a certain threshold may be ignored or discarded. For example, if the threshold is set to 0.75, then audio option $\mathbf{a}_{2,1}(s)$ would be removed, leaving no audio matches for the second video option.

6.2. Audio-video synchronization

When two or more audio options remain for a given video option, it is necessary to perform audio/video synchronization to pair the video option with an appropriate audio option. It may seem that a good choice would be to pick the audio option that has the highest recognition score,

$$k_0 = \arg\max_k a_{j,k}^p(s)$$

but this is not necessarily the best choice for the following reasons. First, it ignores the times at which the au-

[1] The window could be asymmetric, but this adds another parameter, and was found not to be particularly useful.

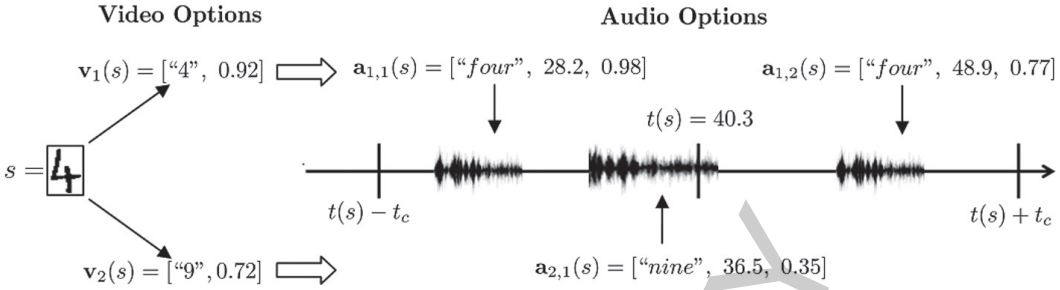


Fig. 4. Audio options, $\mathbf{a}_{j,k}(s)$, for two video options, $\mathbf{v}_1(s)$ and $\mathbf{v}_2(s)$ for the character s.

dio options occur in the audio. As a result, an audio option may be selected that is much further away in time from when the character is written on the board compared to another option with a slightly lower recognition score that corresponds to the correct utterance. This could happen, for example, when there are repeated characters, such as in the equation

$$x + 99y = z$$

The first “nine” that is spoken may have a higher recognition score than the second one and, therefore, may be the one that is selected as the audio option for both characters. Therefore, some alternative methods for synchronization have been considered, and are discussed below [55,57].

6.2.1. Time-difference synchronization

Another simple approach to A/V synchronization is to select the audio option $\mathbf{a}_{j,k}(s)$ that occurs at time, $a^t_{j,k}(s)$, that is closest to the time, $t(s)$, that the video option $\mathbf{v}_j(s)$, is written on the board,

$$k_0 = \arg\min_k |t(s) - a^t_{j,k}(s)|$$

For the example given in Fig. 4, $\mathbf{v}_1(s)$ has two audio options. Since the time of the second audio option is closer to $t(s)$ than that of the first audio option, then $\mathbf{a}_{1,2}(s)$ would be the one that is assigned to $\mathbf{v}_1(s)$. Although this approach is simple, it does not take into account the audio options that are found within a neighborhood of a given option for characters that come before or after it. Therefore, some approaches that are based on the context in which the audio option occurs are presented below.

6.2.2. Neighbor-based methods

To see how context might be used for A/V synchronization, suppose that $\mathbf{v}_j(s)$ is a video option for the character s and let $\mathbf{a}_{j,k}(s)$ be one of its audio options. If $a^t_{j,k}(s)$ is the time at which this audio option occurs, define an *A/V synchronization window* that starts t_b seconds before and ends t_a seconds after this time, i.e.,

$$[a^t_{j,k}(s) - t_b,\ a^t_{j,k}(s) + t_a]$$

With n_a and n_b two positive integers, consider the top video options (those with the largest recognition score) for the n_b characters that occur *before* and the n_a characters that occur *after* the character s. The number of the n_b video options that have an audio option within the A/V window *before* $a^t_{j,k}(s)$ plus the number of the n_a video options that have an audio option within the given window *after* $a^t_{j,k}(s)$ is assigned to the variable $N(\mathbf{a}_{j,k}(s))$. Clearly, this variable may have any value between zero and $n_a + n_b$. The audio option that is then synchronized with the character s is the one that has the largest value of $N(\mathbf{a}_{j,k}(s))$, i.e., $\mathbf{a}_{j,k_0}(s)$ where

$$k_0 = \arg\max_k N(\mathbf{a}_{j,k}(s))$$

If two or more audio options have the same maximum value for $N(\mathbf{a}_{j,k}(s))$, then the one that is the closest in time to $\mathbf{v}_j(s)$ is selected.

As an illustrative example, suppose that the following equation is written on the board,

$$194 + t^2 = x \tag{1}$$

and that the character $s =$ “4” has been classified as ambiguous. In addition, suppose that the first video option for this character is $v^c_1(s) =$ “4” and that $a^c_{1,1}(s) =$ “four” is an audio search term. Over the audio search window that is placed symmetrically around $t(s)$ as illustrated in Fig. 5, note that two occurrences of “four” are found. So the question is “which one corresponds to the video option $\mathbf{v}_1(s)$?” Suppose that $n_a = n_b = 1$

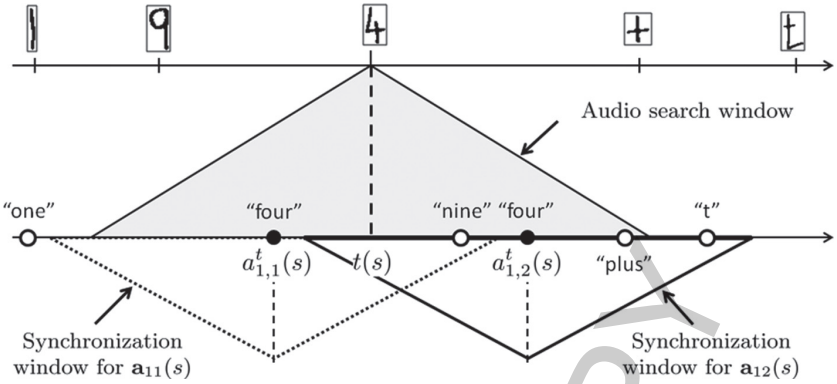


Fig. 5. Illustration of the audio search window for the audio search term "four" and the A/V synchronization windows for two audio options.

and that "9" and "+" are the top *video options* for the character before and after the character s, respectively, i.e., the video options with the highest recognition scores. Within the A/V synchronization window for $\mathbf{a}_{1,1}(s)$, there are no audio options for "nine" *before* time $a^t_{1,1}(s)$ and no audio options for "plus" *after* time $a^t_{1,1}(s)$. Therefore, $N(\mathbf{a}_{1,1}(s)) = 0$. Performing the same search over the A/V synchronization window for the second audio option at time $a^t_{1,2}(s)$, we see that $N(\mathbf{a}_{1,1}(s)) = 2$ since an audio option is found for "nine" within the window before time $a^t_{1,2}(s)$ and one is found for "plus" within the window after time $a^t_{1,2}(s)$. Therefore, $\mathbf{a}_{1,2}(s)$ is the audio option that would be synchronized (paired) with the character $s =$ "4".

6.2.3. Selective neighbor-based methods

In the neighbor-based method for A/V synchronization described above, the top video options of the neighboring characters are assumed to be correct, and the audio search terms for these options are the ones that are used when searching for neighbors and in determining the value of $N(\mathbf{a}_{j,k}(s))$. However, since character recognizers are not perfect, the top video options may be incorrect thereby resulting in poor synchronization. An alternative is to select a subset of the neighboring video options that include those that have the highest probability of being correct. One way to do this is presented below through an illustrative example.

Suppose that Eq. (1) is written on the board, and that an audio option for the character $s =$ "4" is to be found. Shown in Fig. 6 is a video option $\mathbf{v}_1(s)$ for this character along with one of its audio options, $\mathbf{a}_{1,k}(s)$. Also shown are the top video options for the $n_b = 2$ characters *before* and the $n_a = 2$ characters *after* the character s, and the audio options for these characters that are found within the given synchronization window. Using the neighbor-based method with $n_a = n_b = 1$, $N(\mathbf{a}_{1,1}(s))$ would be equal to one instead of two because the video option before the character s is incorrectly recognized as "g" and no utterance of "gee" is found within the given window. However, suppose that $n_a = n_b = 2$, and that out of the two characters before s, the one with the highest recognition score is selected and the same is done for the two characters after s. In this case, $N(\mathbf{a}_{1,1}(s))$ would be equal to two since the incorrectly recognized character would not be used. So, with this approach, in addition to the number of characters before and after s that are considered, two additional parameters are defined, l_b and l_a, that specify how many of the n_b and n_a characters, respectfully, will be selected.

Instead of selecting which neighbors to use based on video scores, the selection may be based on the audio. More specifically, with audio-based neighbor selection, the video options that are selected are those whose audio search terms have the largest number of phonemes. These are the ones that have a higher probability of correctly finding an audio option within the A/V synchronization window, if one exists. As with the video-based neighbor selection, two additional parameters are needed, l_b and l_a, that specify how many of the n_b and n_a characters will be selected.

6.2.4. Rank sum based synchronization

Five different ways to perform audio-video synchronization were presented in the previous sections: score-based, time-difference, neighbor-based, and selective neighbor-based methods using either video or audio. For each synchronization method, the audio options may be rank-ordered, and a rank number assigned to each. For example, if audio option $\mathbf{a}_{j,k}(s)$ is the l^{th} best option using the first synchronization method (recognition score) then the rank, $R_1(\mathbf{a}_{j,k}(s))$,

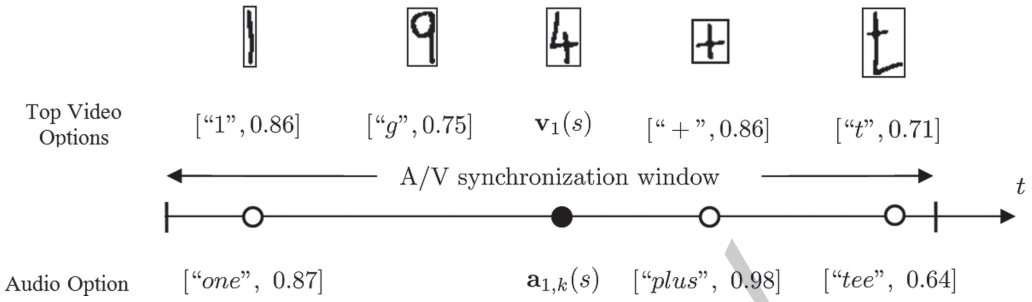


Fig. 6. The audio options that are found within an A/V synchronization window around $\mathbf{a}_{1,k}(s)$ for the top video options of the two characters before and the two characters after the character "4".

for $\mathbf{a}_{j,k}(s)$ using this method would be equal to l. Finding the ranks for this option using the other synchronization methods gives a set of five rank numbers that may be summed to give a rank-sum score. Finding the rank-sum score for each audio option, the one with the lowest score is then selected as the one to be synchronized with $\mathbf{v}_j(s)$, i.e., $\mathbf{a}_{j,k_0}(s)$ where

$$k_0 = \arg\min_k \sum_{i=1}^{5} R_i(\mathbf{a}_{j,k}(s))$$

In the case of a tie, any one of a number of possible tie-breaking strategies may be used, such as selecting the audio option that is the closest in time to when the character s was written on the board.

7. Audio-video combination

The output of the audio-video synchronizer is a set of ambiguous characters that have one or more video options, $\mathbf{v}_j(s)$, for each character s with each video option having only one audio option $\mathbf{a}_{j,k_0}(s)$. For example, shown in Fig. 7 are two video options for the character s. Three audio options are found for $\mathbf{v}_1(s)$ using the audio search term "four" and only one audio option is found for $\mathbf{v}_2(s)$ using the audio search term "nine". The output of the A/V synchronizer for the first video option is denoted by $\mathbf{a}_{1,k_0}(s)$. Since there is only one audio option for $\mathbf{v}_2(s)$, no synchronization (pairing) is required. So, with two audio/video pairs,

$$[\mathbf{v}_1(s), \mathbf{a}_{1,k_0}(s)], \quad [\mathbf{v}_2(s), \mathbf{a}_{2,1}(s)]$$

the last step is to determine which pair is the correct one, thereby leading to the final recognition result, either "4" or "9". This decision is made based on the recognition scores from each recognizer, the audio/video metrics found during synchronization, and perhaps on some other sets of parameters. Similar combination approaches have been used to improve the recognition accuracy of handwriting recognizers [59], speech recognizers [12] and a combination of the two [21]. We considered several rank-level and measurement-level combination techniques [53] such as rank sum and weighted sum rule using classifier-specific weights and character-specific weights as described below.

7.1. Rank based techniques

The challenge often encountered when combining the outputs of two or more classifiers is that the recognition scores are not generally normalized with respect to each other. In other words a score of 8.0 out of 10 for one classifier may not mean the same as a score of 8.0 out of 10 for another. In such situations, a commonly used approach is to use the rank sum (Borda count). With this approach, the various options (or features) for each classifier are assigned a rank [53]. The ranks are then summed for each option and the recognition result having the lowest rank sum is selected as the output. In the case of a tie, a tie-breaking strategy is used. One of the advantages of a rank based technique is that there is no need to assign weights to the audio and video recognizers or to normalize the recognition scores.

An example of the rank sum method is shown in Fig. 8 where the number two is to be recognized. The recognition scores for the video and audio recognizers along with their ranks are shown in the table. Note that if the sum of the recognition scores were used to select the final character, then the number seven would have been the final result whereas the rank sum results in a correct recognition. The relatively high audio recognition score for the audio option corresponding to the character "a" is due to the presence of the phoneme for character "a" that is found in the number eight that oc-

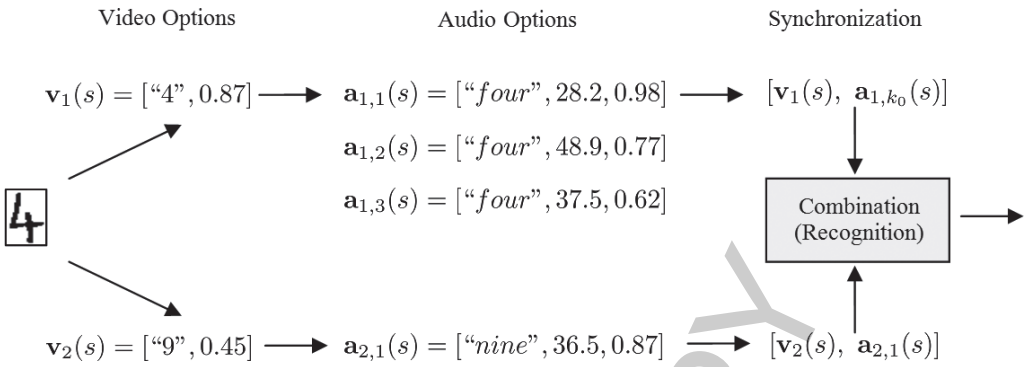
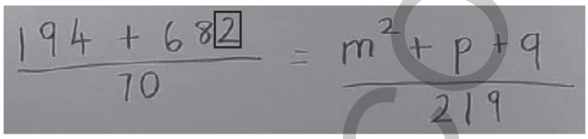


Fig. 7. A character s with two video options, with the first one having three audio options. A/V synchronization selects the best audio option for $\mathbf{v}_1(s)$, and the final step is to decide which audio-video pair is the correct one, leading to the final recognition result.



Video Option	Audio Option	Video Recognition Score and Rank		Audio Recognition Score and Rank		Recognition Score Sum	Rank Sum
$v_j^c(s)$	$a_j^c(s)$	$v_j^p(s)$	R_v	$a_j^p(s)$	R_a	$v_j^p(s) + a_j^p(s)$	$R_v + R_a$
z	z	0.98	1	0.23	4	1.21	5
2	2	0.89	2	0.65	2	1.54	4
a	a	0.82	3	0.51	3	1.33	6
1	1	0.81	4	0.14	5	9.5	9
7	7	0.81	5	0.77	1	1.58	6

Fig. 8. The final recognition step in selection the best audio-video pair for a character. Shown in this example are the results of recognizing the character “2” based on the sum of the audio and recognition scores and on the rank sum score.

curs just before the number two. There is also a high audio recognition score for the audio option “7” because the number seven actually occurs in the audio just after the number two.

7.2. *Weighted sum of recognition scores*

As suggested in the example in Fig. 8, another way to combine the audio and video recognition scores is simply to form the sum [53]. However, given that the video and the audio recognizers may perform differently in their ability to recognize characters, this should be accounted for in the sum. The simplest approach would be to use the same recognizer-specific weights, w_v and w_a, for all characters,

$$z(s) = w_v v_j^p(s) + w_a a_j^p(s)$$

If, for example, the audio recognizer was determined to be much more accurate than the character recognizer, in general, then more weight should be placed on the audio recognition score when selecting the final output. However, since the accuracy of the audio and video text recognizers may be different for each character, then using different weights for each character has the potential to further improve the recognition accuracy. Consider, for example, the number seven. Since “seven” has two phonemes, the audio recognizer will have an easier time recognizing this character compared to single phoneme characters such as “b” and “d”. The video recognizer, on the other hand, will have more difficulty recognizing the number “1” (compared to the letter “l”) than it will have recognizing the number “3”.

One way to assign a weight $w_V(v_j^c(s))$ to the j^{th} video option for the character s is to use the video text recognizer’s accuracy-related metrics, such as precision and sensitivity for the character label $v_j^c(s)$. The value of the audio weight $w_A(v_j^c(s))$ may either be computed in a similar fashion with the weights being

Table 2
Summary of recognition results

Rescoring	Ambiguity detection	Options	AVS	AVC	Rate
–	–	One	–	–	53.7
✓	–	One	–	–	62.2
✓	–	All	✓	✓	61.4
✓	–	$T_O = 0.80$	✓	✓	64.0
✓	$T_R = 0.85$	All	✓	✓	64.5
✓	$T_A = 0.98$	Four	✓	✓	67.6

normalized so that they sum to one, or the weight may be set to $w_A(v_j^c(s)) = 1 - w_V(v_j^c(s))$.

8. Experiments

This section summarizes some of the key results from an extensive set of experiments that were done to test the performance of the various approaches to audio-video character recognition that have been presented here [55]. An attempt was made to isolate the contribution of each step in improving the recognition accuracy of the overall system under a variety of conditions, but this is an extremely difficult task because of the interactions of all of the components and the sheer number of possible combinations of methods introduced. For example, determining what approach is best for option selection will depend on what approaches are used in all of the tasks that follow. Although not presented here, a discussion of the difficulties encountered when there is heavy occlusion by the instructor, poor time-stamping of the video options, and large skews in time between the audio and video may be found in [55]. Here, we summarize the overall performance of the system using what appears to be the best system configuration for the audio and video recognizers that were used. What is significant about the results is not the specific numbers that were obtained for the recognition rates, because they could be improved with any improvement in either recognizer. What is important is the increase that is afforded by incorporating audio into the recognition process, and how the audio is used to achieve this increase.

8.1. Setup: Data set and implementation

The recording equipment used to capture video consisted of a commercially available off-the-shelf video camera (Sanyo VPC-HD1A 720p High-Definition Digital Media Camera) and a wired microphone. The videos were recorded in a classroom-like setting with mathematical content being written on the whiteboard and being spoken by the instructor. The camera was configured to capture video with a resolution of 1280×720 pixels at 30 frames per second. The main data set is organized into two sets, one for training and one for evaluation. The data set has 9,484 characters from two instructors, 4,414 of which are in the training set and 5,070 are in the test set. Sample data sets are available online [9] along with instructions on how to obtain the complete data set.

8.2. Baseline system

To evaluate the effectiveness of the audio-video character recognizer, two baseline systems were used for comparison. The first is the character recognizer used alone, with no assistance from the audio. With this system, the recognition accuracy was 53.7% using the scores from GOCR, and it was 62.2% if these scores were replaced with conditional probabilities as described in Section 5.1. These results are given in the first two rows of Table 2.

The second baseline system is one in which all characters are classified as ambiguous (no ambiguity detection or option selection), with rank sum based synchronization and audio-video combination using recognizer-specific weights $w_v = 0.8$ and $w_a = 0.2$. The character recognition accuracy for this system as shown in the third row of Table 2 was 61.4%. The lower recognition rate here demonstrates the importance of ambiguity detection and option selection.

8.3. Results

After extensive testing, it was found conclusively that instead of using the raw recognition scores from the character recognizer, better overall recognition rates are obtained when they are replaced with estimates of the conditional probability that the character is correctly classified given the raw recognition score as discussed in Section 5.1. It was observed that this rescoring not only resulted in a reordering of the video options so that more characters ended up with the cor-

rect video option with the highest score, but it also resulted in an increase in the number of characters that had the correct video option within the top L scores for any value of L. Therefore, in all of the following results, this rescoring was performed. It is believed that this rescoring should be used for any character recognizer that is used.

In the absence of an ambiguity detector, determining which options are sent to the audio-video recognizer for additional processing may be done in two ways. The first is to select a fixed number of options for each character, and the other is to send only those options that exceed a threshold (or only one option if no options exceed the threshold). Of these two methods, the best was to use an option selection threshold with $T_O = 0.80$, which resulted in a recognition rate of 64.0% as shown in the fourth row of Table 2.

When an ambiguity detector is used, the methods proposed to classify a character as ambiguous include simple and character-specific thresholds. Which type of threshold to use depends on how the ambiguous characters are processed. If there is no option selection criterion (all characters that are classified as ambiguous are sent to the audio recognizer), then a relative threshold value of $T_R = 0.95$ was the best approach, which resulted in a recognition rate of 64.5% as shown in the fifth row of Table 2. This is approximately the same as that obtained when it is assumed that all characters are ambiguous, and the video options that exceed an option selection threshold of $T_O = 0.80$ are sent to the audio-video synchronizer (fourth row of Table 2).

When both an ambiguity detector and an option selection criterion is used, the best recognition rates were achieved using an absolute threshold value of $T_A = 0.98$ for ambiguity detection, and sending four (a fixed number) options to the audio recognizer for each ambiguous character. As shown in Table 2, in this case the recognition rate was 67.6%. What is interesting to note is that for the data set that was used, 54% of the characters were classified as non-ambiguous and for these characters the recognition rate was 85.1%. For the 46% that were classified as ambiguous, 23% of those that would have been incorrectly recognized based on the VTR were corrected by the audio. On the other hand, 1.6% of the characters that would have been recognized correctly were changed to an incorrect character by the audio recognizer.

Each video option that is selected for audio-video combination that has two or more audio options, synchronization (selection of one of the audio options) must be performed. Audio-video synchronization is a difficult task, and how well it can be done is determined by many factors including the accuracy of the video time-stamping (estimation of the time that the character is written on the board), the audio-video alignment (how close in time a character is written to when it is verbalized), and the accuracy of the video and audio recognizers. A summary of the experimental results is as follows. In the case of good video time-stamping (the time difference between the true and estimated times is less than two seconds) and good audio-video alignment (a character is verbalized within four seconds of its being written), each of the synchronization methods performed well, with the feature rank sum based approach performing slightly better. However, in the case of either poor video time stamping or poor audio-video alignment, the synchronization results become worse, with the time difference based techniques performing considerably worse than the others, with the A/V neighbor based and rank sum based techniques performing equally well.

The final step is audio-video combination. In each experiment, ambiguity detection and option selection with a relative threshold of 0.9 was used. With this threshold, 35.7% of the characters are classified as non-ambiguous and 64.3% as ambiguous. The VTR accuracy for the non-ambiguous characters is 79.8% and 39.2% for the ambiguous characters. Since A/V combination processes only the ambiguous characters, the recognition accuracy for these characters should increase. With rank sum based A/V synchronization, the A/V combination results are as follows. With 35.7% of the characters in the test data set classified as ambiguous, without audio the recognition rate for these characters is 39.2%. If the rank sum method is used for audio-video combination, this rate increases to 50.2%, and if recognizer weights of 0.8 and 0.2 are used for the video and audio recognizers, respectively, then this rate increases to 56.9%.

9. Conclusions and future work

This paper presented a number of ways to combine the output of a character recognizer with an audio recognizer to improve the overall recognition accuracy of mathematical equations. The components of the system included a video text recognizer, ambiguity detection, an audio recognizer, audio-video synchronization, and audio-video combination (fusion) of the outputs of the audio and video recognizers. Experiments

conducted over a large data set, consisting of videos recorded in a classroom like environment, demonstrate that significant improvements in character recognition accuracy can be achieved by combining audio with video. While the system made use of specific text and speech recognizers (word spotter), the proposed techniques may be used with other recognizers as well. We are currently in the process of investigating the use of mathematical grammar and the spoken content to improve the structure recognition accuracy associated with handwritten mathematical content. Other avenues for future work include exploring the use of a mathematical grammar for the audio-video based character recognition stage and using time-stamping information along with character recognition results to spot sequences of words instead of single words.

Acknowledgments

This work was supported, in part, by research funds from Chung-Ang University, Seoul, Korea.

References

[1] W. Aly, S. Uchida and M. Suzuki, A large-scale analysis of mathematical expressions for an accurate understanding of their structure, Int. Workshop on Document Analysis Systems, 2008, 549–556.

[2] R. Anderson, C. Hoyer, C. Prince, J. Su, F. Videon and S. Wolfman, Speech, ink, and slides: The interaction of content channels, Int. Conf. on Multimedia, 2004, 796–803.

[3] R.H. Anderson, Two-dimensional mathematical notations. in: Syntactic Pattern Recognition Applications, 1977, pp. 147–177.

[4] L. Anthony, J. Yang and K.R. Koedinger, Evaluation of multimodal input for entering mathematical equations on the computer. CHI '05 extended abstracts on Human factors in computing systems, 2005, pp. 1184–1187.

[5] A.-M. Awal, H. Mouchere and C. Viard-Gaudin, Towards handwritten mathematical expression recognition, Int. Conf. on Document Analysis and Recog, 2009, pp. 1046–1050.

[6] K.-F. Chan and D.-Y. Yeung, Mathematical expression recognition: A survey, *International Journal on Document Analysis and Recognition* **3** (2000), 3–15.

[7] F. Chang, C.-J. Chen and C.-J. Lu, A linear-time component-labeling algorithm using contour tracing technique, *Comput. Vis. Image Understanding* **93**(2) (2004), 206–220.

[8] P.A. Chou, Recognition of equations using a two-dimensional stochastic context-free grammar, SPIE Visual Comm. and Image Processing IV, 1989, 852–863.

[9] Classroom Video Data Set. http://users.ece.gatech.edu/∼smita/dataset/, 2012 (accessed August 20, 2012).

[10] Nuance – Dragon NaturallySpeaking Speech Recognition Software, http://www.nuance.com/naturallyspeaking/, 2012 (accessed August 20, 2012).

[11] R.J. Fateman, T. Tokuyasu, B.P. Berman and N. Mitchell, Optical character recognition and parsing of typeset mathematics, *Journal of Visual Communication and Image Representation* **7** (1996), 2–15.

[12] J. Fiscus, A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER). In IEEE Workshop on Automatic Speech Recognition and Understanding, 1997, 347–354.

[13] G. Friedland, W. Hurst and L. Knipping, Educational multimedia. Multimedia, *IEEE* (2008), 54–56.

[14] P.D. Gader, M.A. Mohamed and J.M. Keller, Fusion of handwritten word classifiers, *Pattern Recogn. Letters* (1996), 577–584.

[15] GOCR. http://jocr.sourceforge.net/, 2012 (accessed August 20, 2012).

[16] L.-W. He, Z. Liu and Z. Zhang, Why take notes? use the whiteboard capture system, IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, 2003, 776–779.

[17] L.-W. He and Z. Zhang, Real-time whiteboard capture and processing using a video camera for remote collaboration, IEEE Transactions on Multimedia, 2007, 198–206.

[18] T.K. Ho, J.J. Hull and S.N. Srihari, Decision combination in multiple classifier systems, *IEEE Trans Pattern Anal Mach Intell* (1994), 66–75.

[19] HTK. http://htk.eng.cam.ac.uk/, 2012 (accessed August 20, 2012).

[20] J. Hunsinger and M. Lang, A single-stage top-down probabilistic approach towards understanding spoken and handwritten mathematical formulas, INTERSPEECH, 2000, pp. 386–389.

[21] J. Hunsinger and M. Lang. A speech understanding module for a multimodal mathematical formula editor, IEEE Inter. Conf. on Acoust., Speech, and Sig. Proc., 2000, 2413 –2416.

[22] A. Jain, K. Nandakumar and A. Ross, Score normalization in multimodal biometric systems, Pattern Recognition, 2005.

[23] X. Jiang, K. Yu and H. Bunke, Classifier combination for grammar-guided sentence recognition, First International Workshop on Multiple Classifier Systems, 2000, 383–392.

[24] S.X. Ju, M.J. Black, S. Minneman and D. Kimber, Summarization of video-taped presentations: Automatic analysis of motion and gesture, IEEE Trans. on Circuits and Systems for Video Technology, 1998, 686–696.

[25] A. Kosmala and G. Rigoll, On-line handwritten formula recognition using statistical methods, In Int. Conf. on Pattern Recognition, 1998, 1306–1308.

[26] A. Kosmala, G. Rigoll, S. Lavirotte and L. Pottier, On-line handwritten formula recognition using hidden markov models and context dependent graph grammars, Int. Conf. on Document Analysis and Recognition, 1999, 107–110.

[27] K. Kurihara, M. Goto, J. Ogata and T. Igarashi, Speech pen: predictive handwriting based on ambient multimodal recognition, SIGCHI conference on human factors in computing systems, 2006, 851–860.

[28] H.-J. Lee and M.-C. Lee, Understanding mathematical expressions using procedure-oriented transformation, *Pattern Recognition* (1994), 447–457.

[29] H. Li, D. Doermann and O. Kia, Automatic text detection and tracking in digital video, IEEE Transactions on Image Processing, 2000, 147–156.

[30] S. Lucey, V. Chandran and S. Sridharan, A theoretical framework for independent classifier combination, Int. Conf. on Pattern Recognition, 2002.

[31] S. Lucey, T. Chen, S. Sridharan and V. Ch, Integration strategies for audio-visual speech processing: Applied to

text-dependent speaker recognition, *IEEE Trans. Multimedia* (2005), 495–506.
[32] C. Malon, S. Uchida and M. Suzuki, Mathematical symbol recognition with support vector machines. Pattern Recognition Letters, 2008, 1326–1332.
[33] S. Medjkoune, H. Mouchère, S. Petitrenaud and C. Viard-Gaudin. Handwritten and audio information fusion for mathematical symbol recognition, Int. Conf. on Document Analysis and Recognition, 2011, 379–383.
[34] SpeechServer – Microsoft Corp. http://www.microsoft.com/SPEECH/, 2012 (accessed Aug. 20, 2012).
[35] Nexidia, http://www.nexidia.com/, 2012 (accessed Aug. 20, 2012).
[36] OpenCV, http://opencv.willowgarage.com/wiki/, 2012 (accessed August 20, 2012).
[37] J.A. Pittman, Handwriting recognition: Tablet pc text input. Computer, 2007, 49–54.
[38] R. Plamondon and S.N. Srihari, On-line and off-line handwriting recognition: A comprehensive survey, IEEE Trans. Pattern Anal. and Machine. Intelligence, 2000, 63–84.
[39] G. Potamianos, C. Neti, J. Luettin and I. Matthews, Audio-visual automatic speech recognition: An overview. In Issues in Visual and Audio-Visual Speech Processing, G. Bailly, E. Vatikiotis-Bateson, and P. Perrier (Eds.), 2004, MIT Press.
[40] D. Prusa and V. Hlavac, Mathematical formulae recognition using 2d grammars, Int. Conf. on Document Analysis and Recognition, 2007, 849–853.
[41] S. Quiniou et al., Hamex – A handwritten and audio dataset of mathematical expressions, Int. Conf. on Document Analysis and Recog., 2001, 452–456.
[42] A.F.R. Rahman, H. Alam and M.C. Fairhurst, Multiple classifier combination for character recognition: Revisiting the majority voting system and its variations, Int. Workshop on Document Analysis Systems, 2002, 167–178.
[43] S. Rossetto, F. Varej ao and T.W. Rauber, An expert system application for improving results in a handwritten form recognition system, Int. Conf. on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, 2002, 383–392.
[44] A. Sánchez, C.A.B. Mello, P.D. Suárez and A. Lopes, Automatic line and word segmentation applied to densely line-skewed historical handwritten document images, *Integr Comput-Aided Eng* (2011), 125–142.
[45] E. Saund, Bringing the marks on a whiteboard to electronic life. In Int. Workshop on Cooperative Buildings – Integrating Information, Organizations, and Architecture, 1999, 69–78.
[46] Y. Shi, H. Li and F.K. Soong, A unified framework for symbol segmentation and recognition of handwritten mathematical expressions. In Int. Conf. on Document Analysis and Recognition, 2007, 854–858.
[47] M. Shridhar, G.F. Houle and F. Kimura, Recognition strategies for general handwritten text documents, Integr. Comput.-Aided Eng., 2009, 299–314.
[48] K. Sirlantzis, S. Hoque and M.C. Fairhurst, Trainable multiple classifier schemes for handwritten character recognition, Int. Workshop on Multiple Classifier Systems, 2002, 169–178.
[49] CMU Sphinx, http://cmusphinx.sourceforge.net/, 2012 (accessed August 20, 2012).
[50] Q. Stafford-Fraser and P. Robinson, Brightboard: A video-augmented environment, SIGCHI Conference on Human factors in Comp. Syst: Common Ground, 1996, 134–141.
[51] L. Tang, Semantic content analysis and user interfaces for instructional video indexing, PhD thesis, Columbia University, 2006.
[52] K. Toyozumi, N. Yamada, T. Kitasaka, K. Mori, Y. Suenaga, K. Mase and T. Takahashi, A study of symbol segmentation method for handwritten mathematical formula recognition using mathematical structure information. In Int. Conf. on Pattern Recognition, 2004, 630–633.
[53] S. Tulyakov, S. Jaeger, V. Govindaraju and D. Doermann, Review of classifier combination methods. In Studies in Computational Intelligence: Machine Learning in Document Analysis and Recognition, 2008, 361–386.
[54] M. Van Erp, L. Vuurpijl and L. Schomaker, An overview and comparison of voting methods for pattern recognition. In Int. Workshop on Frontiers in Handwr. Recog., 2002, 195.
[55] S. Vemulapalli, Audio-Video Based Handwritten Mathematical Content Recognition. PhD thesis, Georgia Institute of Technology, 2012.
[56] S. Vemulapalli and M. Hayes, Grammar-Assisted Audio-Video Equation Recognition, Proc. 18th International Conference on Digital Signal Processing, 2013.
[57] S. Vemulapalli and M. Hayes, Synchronization and Combination Techniques for Audio-video Based Handwritten Mathematical Content recognition in Classroom Videos, Proc. 11th Inter. Conf. on Intelligent Syst. Design and App., 2011.
[58] Embedded Viavoice. en.wikipedia.org/wiki/IBM_Via Voice, 2012 (accessed August 20, 2013).
[59] W. Wang, A. Brakensiek and G. Rigoll, Combination of multiple classifiers for handwritten word recognition, Int. Workshop on Frontiers in Handwriting Recog., 2002, p. 117.
[60] M. Wienecke, G.A. Fink and G. Sagerer, Toward automatic video-based whiteboard reading. Int. Journal on Document Analysis and Recognition, 2005, 188–200.
[61] H. Yang, C. Oehlke and C. Meinel, An automated analysis and indexing framework for lecture video portal. In Advances in Web-Based Learning – ICWL 2012, 285–294.
[62] K. Yu, X. Jiang and H. Bunke, Combining acoustic and visual classifiers for the recognition of spoken sentences. In Int. Conf. on Patt. Recog., 2000, 491–494.
[63] R. Zanibbi, D. Blostein and J.R. Cordy, Recognizing mathematical expressions using tree transformation, *IEEE Trans Pattern Anal Mach Intell* (2002), 1455–1467.