Applying Representative Uninorms for Phonetic Classifier Combination

Gábor Gosztolya[1,⋆] and József Dombi[2]

[1] MTA-SZTE Research Group on Artificial Intelligence
of the Hungarian Academy of Sciences and University of Szeged
Szeged, Hungary
ggabor@inf.u-szeged.hu
[2] University of Szeged, Hungary
dombi@inf.u-szeged.hu

Abstract. When combining classifiers, we aggregate the output of different machine learning methods, and base our decision on the aggregated probability values instead of the individual ones. In the phoneme classification task of speech recognition, small excerpts of speech need to be identified as one of the pre-defined phonemes; but the probability value assigned to each possible phoneme also hold valuable information. This is why, when combining classifier output in this task, we must use a combination scheme which can aggregate the output probability values of the basic classifiers in a robust way. We tested the representative uninorms for this task, and were able to significantly outperform all the basic classifiers tested.

Keywords: uninorms, aggregation function, additive generator function, speech recognition, classifier combination, phoneme classification.

1 Introduction

In Artificial Intelligence, perhaps the most intensively investigated area is that of machine learning in general, and classification in particular. In it the goal is to assign one of the pre-defined class labels to a given example. One of the many areas where classification techniques are applied is that of speech recognition, where small acoustic portions of speech have to be identified as phonemes. Historically, several methods were applied in this given task. First Gaussian Mixture Models (GMMs [9]) were employed, especially due to their good probability estimation capabilities; but their low computational complexity also played a role in their popularity. Later Artificial Neural Networks (ANNs [3]) were used, and with the discovery of Deep Neural Networks (DNNs [12,20]) they have become

⋆ This publication is supported by the European Union and co-funded by the European Social Fund. Project title: Telemedicine-oriented research activities in the fields of mathematics, informatics and medical sciences. Project number: TÁMOP-4.2.2.A-11/1/KONV-2012-0073. This study was also partially supported by TÁMOP-4.2.1/B-09/1/KONV-2010-0005.

V. Torra et al. (Eds.): MDAI 2014, LNAI 8825, pp. 182–191, 2014.

the dominant method. In spite of this, other methods such as Support-Vector Machines (SVM [19]) and AdaBoost [18] were also tested.

For a given task the accuracy of a classification method can usually be characterized by one numerical value. However, experiments show that the sets of patterns misclassified by the different classifiers do not necessarily overlap, so a good combination of the original classifiers may reinforce their strong points. This may account for the interest in classifier combination in several areas of Artificial Intelligence (e.g. [17,22,2]), and also in speech recognition [10].

Speech recognition and phoneme classification in particular, however, have another requirement for classifier combination. Here, besides finding the correct class label (phoneme), the estimated probability value of each possible class label is also important, so a method is required that can aggregate the output likelihood values of the basic classifiers into one likelihood value. It is logical to look for operators like this in the field of fuzzy logic; yet fuzzy logic has quite a big range of possible operators. It would also be logical to expect this operator to output high values if most of its input values are high, and from mostly low scores it should produce a score that is close to zero. Uninorms are a set of operators that have these kind of properties, so next we will examine them.

2 Representable Uninorms

The aggregative operators were first introduced in [5] by selecting a set of minimal concepts that must be fulfilled by an evaluation-like operator. In 1982, Dombi [5] defined the aggregative operator in the following way:

Definition 1. *An aggregative operator is a function* $a : [0,1]^2 \to [0,1]$ *with the properties:*

1. *Continuous on* $[0,1]^2 \backslash \{(0,1),(1,0)\}$
2. $a(x,y) < a(x,y')$ *if* $y < y'$, $x \neq 0$, $x \neq 1$
   $a(x,y) < a(x',y)$ *if* $x < x'$, $y \neq 0$, $y \neq 1$
3. $a(0,0) = 0$ *and* $a(1,1) = 1$ *(boundary conditions)*
4. $a(x,a(y,z)) = a(a(x,y),z)$ *(associativity)*
5. *There exists a strong negation operator* η *such that* $a(x,y) = \eta(a(\eta(x),\eta(y)))$ *(self-DeMorgan identity) if* $\{x,y\} \neq \{0,1\}$ *or* $\{x,y\} \neq \{1,0\}$
6. $a(1,0) = a(0,1) = 0$ *or* $a(1,0) = a(0,1) = 1$

The definition of uninorms, originally given by Yager and Rybalov [21] in 1996, is the following:

Definition 2. *A uninorm* U *is a mapping* $U : [0,1]^2 \to [0,1]$ *that has the following properties:*

1. $U(x,y) = U(y,x)$ *(commutativity)*
2. $U(x_1,y_1) \geq U(x_2,y_2)$ *if* $x_1 \geq x_2$ *and* $y_1 \geq y_2$ *(monotonicity)*
3. $U(x,U(y,z)) = U(U(x,y),z)$ *(associativity)*
4. $\exists \nu_* \in [0,1]\ \forall x \in [0,1]\ U(x,\nu_*) = x$ *(neutral element)*

Uninorms are a generalization of t-norms and t-conorms. By adjusting the value of its neutral element ν_*, a uninorm is a t-norm if $\nu_* = 1$ and a t-conorm if $\nu_* = 0$. The main difference between the definition of aggregative operators and uninorms is that the self-DeMorgan identity requirement does not appear in uninorms, and the neutral element property is not in the definition for the aggregative operators. Fodor [11] showed that uninorms which are strict and continuous on $[0,1] \times [0,1] \backslash (\{0,1\},\{1,0\})$ (also called *representative uninorms*) are equivalent to aggregative operators, and they can be represented by the *additive generator* function g as $U(x,y) = g^{-1}(g(x)+g(y))$. Next, we will briefly explain the application of representative uninorms in expert opinion aggregation, following the work of Dombi [8].

Theorem 1. *Let g be an additive generator of an aggregative operator (i.e. representative uninorm) and consider $\nu_* \in (0,1)$; then $a_{\nu_*} : [0,1]^2 \to [0,1]$ defined by*

$$a_{\nu_*}(x,y) = g^{-1}\big(g(x) + g(y) - g(\nu_*)\big) \tag{1}$$

is an aggregation operator (i.e. representative operator) with neutral element ν_*. The extension to n arguments is given by the formula

$$a_{\nu_*}(\mathbf{x}) = g^{-1}\left(g(\nu_*) + \sum_{i=1}^{n}\big(g(x_i) - g(\nu_*)\big)\right). \tag{2}$$

With this, we can construct an aggregative operator from any given generator function that has the desired neutral value. For example, for the Dombi operator case we get

$$a_{\nu_*}(\mathbf{x}) = \frac{1}{1 + \frac{1-\nu_*}{\nu_*}\prod\limits_{i=1}^{n}\left(\frac{1-x_i}{x_i}\frac{\nu_*}{1-\nu_*}\right)}. \tag{3}$$

By weighting each parameter with a w_i factor ($0 \le w_i \le 1$), we get the general form in the additive case:

$$a_{\nu_*}(\mathbf{w},\mathbf{x}) = g^{-1}\left(\sum_{i=1}^{n} w_i g(x_i) + \big(1 - \sum_{i=1}^{n} w_i\big) g(\nu_*)\right). \tag{4}$$

In general, a weighted aggregative operator lacks associativity and commutativity and it is not a representable uninorm. Note that ν_* can be treated as the $n+1$th input, in which approach the $n+1$ weights sum up to one. Therefore, if the original n weights sum up to one, we get the following, simplified form:

$$a_{\nu_*}(\mathbf{w},\mathbf{x}) = g^{-1}\left(\sum_{i=1}^{n} w_i g(x_i)\right). \tag{5}$$

In the Dombi operator case, we get

$$a_{\nu_*}(\mathbf{w},\mathbf{x}) = \frac{1}{1 + \frac{1-\nu_*}{\nu_*}\prod\limits_{i=1}^{n}\left(\frac{1-x_i}{x_i}\frac{\nu_*}{1-\nu_*}\right)^{w_i}}, \tag{6}$$

or

$$a_{\nu_*}(\mathbf{w},\mathbf{x}) = \frac{\nu_*(1-\nu_*)^{\sum_{i=1}^{n} w_i} \prod_{i=1}^{n} x_i^{w_i}}{\nu_*(1-\nu_*)^{\sum_{i=1}^{n} w_i} \prod_{i=1}^{n} x_i^{w_i} + (1-\nu_*)\nu_*^{\prod_{i=1}^{n} w_i} \prod_{i=1}^{n} (1-x_i)^{w_i}}. \quad (7)$$

If $\sum w_i = 1$, then we get

$$a_{\nu_*}(\mathbf{w},\mathbf{x}) = \frac{\prod_{i=1}^{n} x_i^{w_i}}{\prod_{i=1}^{n} x_i^{w_i} + \prod_{i=1}^{n} (1-x_i)^{w_i}}. \quad (8)$$

Using these formulas we can readily aggregate expert probability values [8]. That is, given the probability vector $\mathbf{x}$ (the opinions of experts) as input and their weight vector $\mathbf{w}$, we can readily calculate the resulting likelihood $a_{\nu_*}(\mathbf{w},\mathbf{x})$. Performing this for all options (classes), we get one likelihood score for each of them, and we can base our decision on the values of this vector.

3 The Phoneme Classification Task

Speech recognition seeks to transcribe audio data; that is, given an audio recording (*utterance*), we would like to find its correct textual representation. This is not a pure classification task, as both the input and the output are of variable length. To overcome this problem, the input utterance A is usually divided into small, equal-sized parts (*frames*) typically 10ms in length; that is, $A = a_1, a_2, \ldots, a_n$. The a_i frames can be classified into one of the previously defined (and language-dependent) phonemes, using specific features extracted from the frame and its near neighbourhood. For the phonetic labels $ph_1, ph_2, \ldots ph_m$, classification produces the likelihood values $P(ph_i|a_j)\ \forall\ 1 \leq i \leq m, 1 \leq j \leq n$, where

$$\sum_{i=1}^{m} P(ph_i|a_j) = 1 \quad (9)$$

holds for all $1 \leq j \leq n$. In the next step, a search is performed based on these posterior probability values, where the intention is to find the most probable *phoneme sequence*. Assuming that the neighbouring frames are independent, we basically look for a phoneme sequence such that the product of the appropriate frame-level posterior probability values is maximal. (In the actual implementation there are a number of restrictions on the allowed phoneme sequences, which were omitted for sake of clarity. We also did not discuss the incorporation of other aspects like phonetic and language models.) The output of this process is the optimal phoneme sequence, from which we remove duplicate neighbouring phonemes.

Although standard classification algorithms are used for the classification of these small speech excerpts, it is not a pure classification task, as we are not only interested in the resulting class label, but also the posterior probability values contain valuable information. (For example, a few incorrectly classified frames can be corrected if the probability value of the surrounding frames for the given phoneme is sufficiently high.) Despite there being a clear connection between the posterior scores and the class labels (e.g. we normally choose the class which has the highest posterior score), it is common to have a classification method that produces accurate class labels, but supplies only inaccurate posterior estimates (e.g. AdaBoost.MH).

3.1 Aggregating Phoneme Classifiers

Classifier combination can be easily incorporated into this scheme as well: instead of relying on the output of a single classification method, we will treat the output of the aggregated probability value as the $P(ph_i|a_j)$ posterior scores. For this, practically any aggregation method can be used. However, it should have parameters that allow us to fine-tune its behaviour to best suit the problem, but avoid having too many parameters, as it would make it next to impossible to set them all properly. Furthermore, as we want to aggregate the likelihood values from sources (i.e. classifiers) of different quality, it would also be nice if we could weight the independent sources. All these points suggest that it would be worth trying representable uninorms, which supply values in the $[0, 1]$ interval as results. The input classifiers can be weighted via the w_i values; and, depending on the g additive generator function used, they may have one or more parameters.

4 Experiments and Results

Having described the phoneme classification problem and representable uninorms, we will now turn to the testing part. First we describe the speech recognition environment, then describe the optimisation process, and finally we present and analyse our test results.

4.1 The Speech Recognition Environment

We used the English TIMIT dataset commonly used for speech recognition experiments [16], with its conventional splitting into training and (core) test sets. We separated 176 utterances from the training set to form a separate development set, and used the standard, 61-long set of phonemes. Phonetic accuracy was measured by applying the edit distance-based accuracy metric, traditionally employed in speech recognition.

We tested three kinds of classifiers, namely Artificial Neural Networks (ANNs [3]) with our custom implementation; Support-Vector Machines (SVM [19]), using the tool LibSVM [4]; and AdaBoost.MH, using the `multiboost` library [1]. The first

Table 1. The accuracy scores got for the three different basic classifiers, and for their combinations

Method		Dev. set	Test set
Basic methods	ANN	77.26%	73.99%
	SVM	75.03%	72.49%
	AdaBoost	75.41%	73.01%
Uninorms	Product	78.54%	74.98%
	Dombi t-norm	78.45%	74.74%
	Generalized Dombi t-norm	78.69%	75.19%

two methods produced well-balanced posterior scores by default, while we calibrated the output of AdaBoost first to have the same standard deviation as those for the ANNs, then normalised them to sum up to one. The neural network had 2000 neurons in its hidden layer, and utilized the sigmoid activation function; we used the RBF kernel in the SVM case; and AdaBoost was trained using 8-leaved decision trees as base learners.

We tested two combination configurations. In the first one, we applied a model of all three types of basic classifiers to find out how effectively we could combine classifiers that were of different types. In the second set of tests we trained three neural networks in the same way, and combined them; with this set of tests we wanted to see how effectively three similar algorithms could be combined. Although the neural networks were trained in the same way, they were not identical due to their random weight initialisation.

4.2 Optimisation

We tested the additive generator functions of three triangular norms. First, as the product operator is also a t-norm, we used its additive generator function, $-\log x$; then we experimented with the one-parametric version of the Dombi t-norm [6], and with the Generalized Dombi Operator that has two parameters [7]. These triangular norm families were chosen based on previous experiences in the field of speech recognition (e.g. [14,13,15]), where we found these norms to be flexible, yet robust. Naturally, many other norms can be used for classifier combination, but this time our aim was not a full-scale comparison of the possible generator functions.

As there were only a few parameters, we did not use a heavyweight optimisation method, but only generated random weights for each classifier and parameters for the generator functions (where necessary) instead. The weights of the classifiers were normalised so as to add up to one; this process was repeated 10000 times, and the best parameter set was chosen.

As is typical in speech recognition, we trained our classifiers on the training set, and evaluated them on the development set. We determined the parameters of the representative uninorms on this development set, then we evaluated the optimal parameter set on the test set to see how robust our achievement was.

Table 2. The accuracy scores got for the three neural networks, and for their combinations

Method		Dev. set	Test set
Basic methods	ANN #1	77.26%	73.99%
	ANN #2	77.11%	74.16%
	ANN #3	77.32%	74.32%
Uninorms	Product	78.54%	75.60%
	Dombi t-norm	78.42%	75.30%
	Generalized Dombi t-norm	78.71%	76.04%

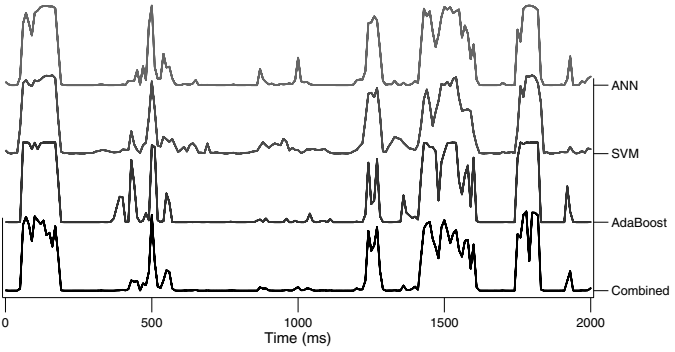

Fig. 1. The input probability values (gray) and the optimal combined values for the uninorm based on the generator of the product norm for the "sil" (silence) phoneme

4.3 Results

The results got for the different basic classifiers can be seen in Table 1. The accuracy scores of the basic classifiers varied significantly both on the development and on the test sets, but ANNs produced the best results. Nevertheless, their combination using the representative uninorms outperformed the base classifiers in each case. It may seem surprising that by using the additive generator function of the Dombi t-norm, our results did not surpass those of the product norm; this can be explained, however, by the fact that we kept the value of the neutral element ν_* at its default value of 0.5, where this behaviour can be anticipated. However, when using the additive generator function of the Generalized Dombi Operator, we were able to significantly outperform even this score; and all these improvements could also be carried over to the test set. (Note that the improvement may not seem that much; but it is indeed significant by speech recognition standards, especially when no language model at all was used.)

When we tried to combine the three, quite similar neural networks (see Table 2), we got very similar results, except that the scores of the basic classifiers varied less this time, especially on the development set. The combined values outperformed the basic classifiers in every case, and although the one using the additive generator of the product norm again performed slightly better than the

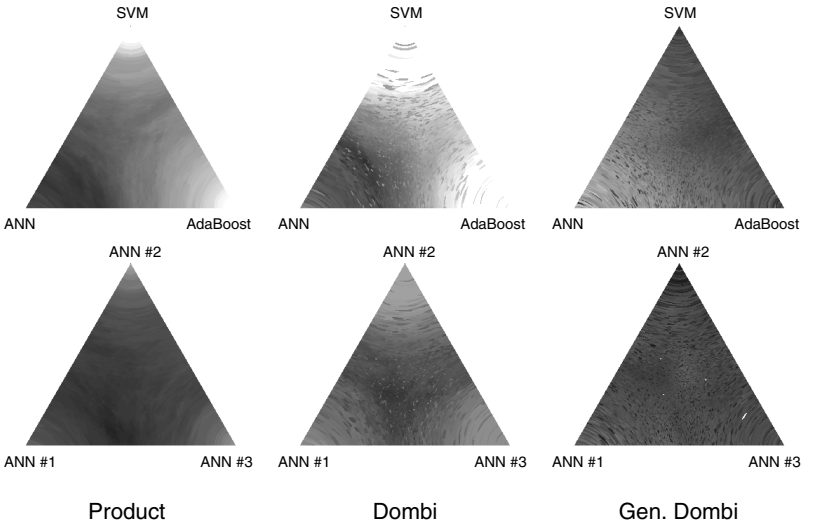

Fig. 2. The accuracy scores got on the development set as a function of classifier weights, for the three different (up) and the three neural network (down) classifier cases. The distance between a point and a corner of the triangle is inversely proportional to the appropriate weight; a darker colour means a higher accuracy score.

other one using the generator function of the Dombi t-norm, the Generalized Dombi Operator again proved to be the best. Note that although this representative uninorm performed similarly on the development set in both cases (78.69% and 78.71%, different basic classifiers and neural networks, respectively), the difference was much bigger on the test set (75.19% and 76.04%). Figure 2 is a plot of the accuracy scores obtained as a function of the weight values. The images belonging to the product norm are smoother, which is due to the additional parameter(s) of the additive generator functions. In the case of the three different classifiers, to achieve high accuracy we should set the weight of the ANNs to a quite large value, but well below 1.0. The exception is the Generalized Dombi Operator case, where the optimal region has roughly equally high weights for SVM and AdaBoost, and a much lower weight for ANNs. For the three neural networks case, not surprisingly, the highest accuracy scores lie in regions where the three weights are roughly equal. This tendency can be seen a bit in the Generalized Dombi Operator case, which probably means that it is more important to properly set the two parameters of the generator function than the classifier weights.

5 Conclusions

The phoneme recognition task of speech recognition is unusual for classifier combination as here not only the correct class label (phoneme) has to be identified

based on the output of the individual classifiers, but its likelihood score also has be determined in a robust way. We chose to test representable uninorms in this task because of their useful properties: they are able to output values in the $[0, 1]$ range, they can handle the weighting of the base classifiers, and – depending on the generative function used – they can have additional parameters to control their behaviour. We tested three types of additive generator functions, namely the product, the Dombi and the Generalized Dombi Operators, and we were able to outperform the base classifiers significantly in each case. We think that even better accuracy scores can be attained by properly setting the value of the neutral element ν_*, which we plan to investigate in the near future.

References

1. Benbouzid, D., Busa-Fekete, R., Casagrande, N., Collin, F.D., Kégl, B.: MultiBoost: a multi-purpose boosting package. Journal of Machine Learning Research 13, 549–553 (2012)
2. Bi, Y., Bell, D.A., Wang, H., Guo, G., Greer, K.: Combining multiple classifiers using Dempster's rule of combination for text categorization. In: Torra, V., Narukawa, Y. (eds.) MDAI 2004. LNCS (LNAI), vol. 3131, pp. 127–138. Springer, Heidelberg (2004)
3. Bishop, C.: Neural Networks for Pattern Recognition. Clarendon Press, Oxford (1995)
4. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2, 1–27 (2011)
5. Dombi, J.: Basic concepts for a theory of evaluation: the agregative operator. European Journal of Operational Research 10, 282–293 (1982)
6. Dombi, J.: A general class of fuzzy operators, the De Morgan class of fuzzy operators and fuzziness measures induced by fuzzy operators. Fuzzy Sets and Systems 8, 149–163 (1982)
7. Dombi, J.: Towards a general class of operators for fuzzy systems. IEEE Transaction on Fuzzy Systems 16(2), 477–484 (2008)
8. Dombi, J.: Bayes theorem, uninorms and aggregating expert opinions. In: Bustince, H., Fernandez, J., Mesiar, R., Calvo, T. (eds.) Aggregation Functions in Theory and in Practise. AISC, vol. 228, pp. 281–291. Springer, Heidelberg (2013)
9. Duda, R., Hart, P.: Pattern Classification and Scene Analysis. Wiley & Sons, New York (1973)
10. Felföldi, L., Kocsor, A., Tóth, L.: Classifier combination in speech recognition. Periodica Polytechnica, Electrical Engineering 47(1), 125–140 (2003)
11. Fodor, J., Yager, R.R., Rybalov, A.: Structure of uninorms. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 5(4), 411–427 (1997)
12. Glorot, X., Bordes, A., Bengio, Y.: Deep sparse rectifier networks. In: Proceedings of AISTATS, pp. 315–323 (2011)
13. Gosztolya, G., Dombi, J., Kocsor, A.: Applying the Generalized Dombi Operator family to the speech recognition task. Journal of Computing and Information Technology 17(3), 285–293 (2009)
14. Gosztolya, G., Kocsor, A.: Using triangular norms in a segment-based automatic speech recognition system. International Journal of Information Technology and Intelligent Computing (IT & IC) (IEEE) 1(3), 487–498 (2006)

15. Kocsor, A., Gosztolya, G.: Application of full reinforcement aggregation operators in speech recognition. In: Proceedings of the 2006 Conference of Recent Advances in Soft Computing (RASC), Canterbury, UK (2006)
16. Lamel, L., Kassel, R., Seneff, S.: Speech database development: Design and analysis of the acoustic-phonetic corpus. In: DARPA Speech Recognition Workshop, pp. 121–124 (1986)
17. Plessis, B., Sicsu, A., Heutte, L., Menu, E., Lecolinet, E., Debon, O., Moreau, J.V.: A multi-classifier combination strategy for the recognition of handwritten cursive words. In: Proceedings of ICDAR, pp. 642–645 (1993)
18. Schapire, R.E., Singer, Y.: Improved boosting algorithms using confidence-rated predictions. Machine Learning 37(3), 297–336 (1999)
19. Schölkopf, B., Platt, J., Shawe-Taylor, J., Smola, A., Williamson, R.: Estimating the support of a high-dimensional distribution. Neural Computation 13(7), 1443–1471 (2001)
20. Tóth, L.: Convolutional deep rectifier neural nets for phone recognition. In: Proceedings of Interspeech, Lyon, France, pp. 1722–1726 (2013)
21. Yager, R.R., Rybalov, A.: Uninorm aggregation operators. Fuzzy Sets and Systems 80(1), 111–120 (1996)
22. Yu, K., Jiang, X., Bunke, H.: Lipreading: A classifier combination approach. Pattern Recognition Letters 18(11-13), 1421–1426 (1997)