Cut the Carp! Using Context to Disambiguate Similar Signals Using Conceptors

Bachelor's Project Thesis

Satchit Chatterji, S3889807, s.chatterji.1@student.rug.nl
Supervisors: Guillaume Pourcel & Prof Dr Herbert Jaeger

Abstract: Humans robustly and ubiquitously use symbolic context to distinguish between noisy or ambiguous percepts. However, no consensus exists on how to properly integrate top-down/symbolic and bottom-up/sub-symbolic information flow in artificial neural networks. One approach to this 'bidirectional information flow' problem in recurrent neural networks (RNNs) is the *conceptor* – a matrix that characterises the linear space of neuronal activations in response to a signal. These may be abstractly combined using Boolean logical connectives. However, they may alternatively be interpreted as *fuzzy* operators, and conceptors themselves corresponding to fuzzy sets. The current work introduces a bidirectional classification model that allows for top-down information to affect bottom-up processes in an RNN using conceptors. Building on conceptor-based classification, three biasing methods are discussed, based on (i) crisp symbols, (ii) probability-quantified uncertainty, and (iii) fuzzy membership-quantified uncertainty. As a demonstration, phonemes are transcribed using the TIMIT dataset. With the goal of using contextual information to help disambiguate similar percepts, the discriminatory power of these biasing paradigms are studied with respect to similar-sounding phonemes.

1 Introduction

It is well studied in cognitive psychology that the human ability to robustly perceive sensory input relies on 'bottom-up' processing, from raw signals to abstract symbolic representations, being influenced by 'top-down' communication of context-relevant information – abstract symbols influencing the perception of signals (Bermúdez, 2014; Mittal et al., 2020). Bottom-up processes include functions such as visual or auditory processing (from neuronal excitations caused by light or sound signals respectively), and top-down influences often arise from patterns in working memory or past experiences.

As an example of top-down context influencing bottom-up processing, the title of this thesis begins with the phrase '*Cut the carp!*', with the final word *carp* referring to several species of freshwater fish. However, those familiar with idiomatic written English would have likely first read the similar phrase '*Cut the crap!*', which is a rude way of asking someone to 'get to the point'. For regular users of colloquial English (apart from, perhaps, fish mongers), the prior sequence of words is unlikely to be encountered as frequently as the latter. Thus, along with one's past experience, the symbolic, top-down context provided by the first two words '*cut*' and '*the*', and the general resemblance of the shapes of the words themselves, we are biased to mispercieve the visual signal 'carp' as 'crap' instead.

Though many such tasks involving reading, speech perception, and object recognition rely on this bidirectional information flow, the importance of the influence of top-down upon bottom-up signals (and vice-versa) in humans is still a debated topic (Rauss & Pourtois, 2013). Additionally, an effective method of implementing this two-way flow in *artificial* decision-making systems is yet unknown. In the realm of artificial decision-making agents, bottom-up processing helps a system to be flexible and general, top-down processing helps disambiguate noisy or ambiguous raw data. *Good-old-fashioned-AI* techniques were largely based on symbolic reasoning, with systems crafted by human experts in the form of decision trees or formal logic (Boden, 2014). In contrast, early deep-learning (DL) models aimed to learn purely from statistical relations in data, and were designed to enable the use of a hierarchy of increasingly abstract feature representations (e.g. Bengio et al., 2006; Krizhevsky et al., 2012). This, more bottom-up approach, has greatly influenced DL methods today.

Though attempts at creating bidirectional learning models exist, they have received less attention from the academic community than DL models such as feed-forward neural networks (Mittal et al., 2020). This project aims to be a first step towards creating a bidirectional classification paradigm in reservoir networks (a subset of recurrent neural networks) using conceptors (Jaeger, 2014). Top-down, symbolic information is used to guide future, bottom-up predictions to reduce the ambiguity between perceptually similar signals. Three related methods to do this are discussed, inspired by different interpretations of human reasoning: symbolic, probabilistic and fuzzy.

Overview of Introduction: Section 1.1 lays out phoneme transcription as an specific case of a bidirectional task. Section 1.2 gives a brief overview of reservoirs and conceptors, and section 1.3 describes how they can be used for classification tasks. Section 1.4 describes a Boolean logic framework for them, and sections 1.5 and 1.6 respectively speak about fuzzy logic and set theory and how conceptor logic relates to it. Next, section 1.7 defines the aperture adaptation operation on conceptors. Section 1.8 describes Jaeger's ∇-rule to compute 'optimal' apertures. Finally, section 1.9 proposes a general framework within which aperture adaptation may act as an uncertainty operator on conceptors, and section 1.10 condenses the previous ideas into a general biasing framework for reservoirs.

1.1 Bidrectional Phoneme Transcription

A phoneme is the smallest unit of sound whose replacement in a given spoken word changes the perception of what the word is itself (Huang et al., 2001). For example, the words 'money' and 'honey' are perceived as different words only because of their respective differing starting sounds, /m/ and /h/[1]. When transcribing a phoneme in isolation, it can be difficult to classify it into a specific class accurately due to signal similarities, for example, the /m/ and /n/ sounds. However, using previously acquired knowledge and the context of the surrounding phonemes that make up the word being spoken, the task becomes simpler (Gianakas & Winn, 2016). Humans would expect that the word 'candy' is more likely to be spoken than 'camdy', and thus the transcription[2] would look like /k ae n d iy/ instead of /k ae m d iy/. Thus, context is able to disambiguate similar-sounding phonemes.

Some previous attempts at 'bidirectional' phoneme classification such as Graves & Schmidhuber (2005) used bi-directional long short-term memory (BLSTM) networks. However, this bidirectionality is different than the one studied here. For BLSTMs, there are two parallel regular LSTM layers where the inputs feed into the network forward in time for one, and backward in time for the other. Thus, in the backward layer, there is presumably a flow of sub-symbolic activation data relating to the previous phonemes. Thus, it is bi-directional in time. This project aims to combine two sets of information – data gathered at a symbolic level and the low-level excitations of a neural network. Though it is also based on previous phonemes, this sort of 'bidirectionality' refers more to this symbolic-subsymbolic divide.

Interestingly, in a phenomenon known as the Ganong effect, humans also use recent signal inputs to disambiguate or 'correct' phoneme recognition that occurred further in the past (Ganong, 1980; Gwilliams et al., 2018; Gianakas & Winn, 2016). A human is much more likely to transcribe the phoneme /k/ rather than /g/ if it occurs before the phonemes /i s/ (i.e. making the word 'kiss'), and likewise more likely to transcribe /g/ rather than /k/ if it is followed by /i f t/ (to make the word 'gift'). Though implementing the Ganong effect in an automatic transcription system would presum-

[1] A phoneme is generally notated by placing a representative symbol in between a pair of forward-slashes, which looks like: '/aa/'. When transcribing a word with multiple phonemes, all phonemes are placed between one pair of forward-slashes, for example the word 'symbol' would be transcribed as '/s ih m b el/'.

[2] All example in-text phonemic transcriptions used in this thesis were taken from the TIMIT dataset (Garofolo et al., 1993). Note that these symbols differ from the more commonly recognised International Phonetic Alphabet.

ably assist in making more accurate predictions, the current research was designed with the intent of being a true 'online' transcription task, i.e. every new input is expected to have an immediate corresponding prediction without waiting for any more information. Thus, only previous signal information is available with respect to the current input. The Ganong effect, in comparison, cannot be implemented in a true online task – older predictions would need to be amendable by the system, or there would need to be a delay in the predictions with respect to the input.

1.2 Conceptors: An Overview

Motivated by the need of marrying high-level symbolic concepts and the data-based dynamical inner workings of a recurrent neural network[3] (RNN), conceptors were introduced by Jaeger (2014) as a matrix representing a linear subspace of the activations of such an RNN given a driving signal. As such, they act as abstract, symbol-like objects that represent a signal, while also being intimately bound with the low-level excitations of the neurons in the reservoir. They may be symbolically joined using an analogue of a Boolean logic system, and ordered to form 'sub-conceptor' relationships that may be seen to correspond with symbolic hierarchies in the data itself. The details in the rest of this section borrow from the description of conceptors given in Jaeger (2014) and Mossakowski et al. (2018).

1.2.1 Reservoirs

Though conceptors have been extended to other artificial neural network architectures (such as feed-forward neural networks in He & Jaeger, 2018), a conceptor *here* is defined in terms of an RNN whose weights do not change after random initialisation – rather, these neurons act as a *reservoir* of recurrent neural connections.

Let us now assume a reservoir is initialised as having N neurons, the strength of whose recurrent neural connections can be represented by a *connection matrix* or *weight matrix* W of size $N \times N$, where the entry $w_{i,j}$ is the weight of the connection between neuron i to neuron j. Let the input signal or pattern at time t be denoted by $s(t)$, which is a vector of M dimensions. Let W^{in} be the weights that connect the input with each neuron in the reservoir – thus, W^{in} is a matrix of size $N \times M$. Finally, let b be an N-dimensional bias vector that contains time-invariant offsets added into each neuron. Thus, given an arbitrary state of the excitation of neurons at time $t = 0$, while driving the signal through the reservoir over time we get the following *state-update rule*:

$$x(t+1) = \tanh(Wx(t) + W^{in}s(t+1) + b) \quad (1.1)$$

where *tanh* is the hyperbolic tangent. The vector $x(t)$ thus contains the excitation of the neurons. If this pattern drives the reservoir for a total of T time steps, the network states are approximately confined to a linear subspace of the total space of neuronal activations which is pattern-dependent. The states may be concatenated into an excitation collection matrix X of shape $N \times T$ of the form $[x(1)\ x(2)\ ...\ x(T)]$. This can be interpreted as the sequence of states in N-dimensional reservoir state space that are visited by the neuronal activations for this signal. This is also referred to as a *trajectory.*

We may study how much the states correlate with one another in time by computing a state correlation matrix as:

$$R = XX'/T \quad (1.2)$$

The shape of R is $N \times N$, and the entry $r_{i,j}$ is the correlation of $x(n)_i$ with $x(n)_j$ over time, i.e. how closely the neurons' states correlate with one another.

1.2.2 Computing Conceptors

A conceptor is a matrix that represents the linear subspace of the trajectory of reservoir activations. It can be computed using the state correlation matrix as:

$$\begin{aligned} C(R,\alpha) &= R(R+\alpha^{-2}I)^{-1} \\ &= (R+\alpha^{-2}I)^{-1}R \end{aligned} \quad (1.3)$$

where I is the identity matrix of shape $N \times N$, and $\alpha \in (0, \infty)$ is a scaling factor known as the *aperture* (more on apertures can be found in section 1.7). The conceptor may be visualised as an

[3]A recurrent neural network is a kind of artificial neural network that has at least a subset of its neurons connected recurrently – there are loops in the information flow.

N-dimensional hyper-ellipse whose axes' lengths in each direction correspond to the singular value in that direction.

The state correlation matrix can also be recomputed using a conceptor and an aperture as:

$$\begin{aligned} R(C,\alpha) &= \alpha^{-2}(I-C)^{-1}C \\ &= \alpha^{-2}C(I-C)^{-1} \end{aligned} \tag{1.4}$$

A conceptor matrix thus is real, symmetric, and positive semi-definite with singular values in the range of $[0, 1]$.

1.3 Conceptor Classification

Most modern classification paradigms in machine learning are trained to be *discriminative* classifiers – they classify by looking at the differences between learned input patterns. Thus, to learn to discriminate a *new* pattern, differences between the old classes and new classes need to be learnt, often requiring a complete re-training or fine-tuning. However, conceptor classification as was described by Jaeger (2014) is based on what he called 'pattern-local' classification, where patterns are incrementally learned, and classification is done purely with respect to comparing an input with each learned class. This classification scheme is recreated here and will be referred to as the 'classical' conceptor classification scheme.

Conceptor Creation (Training Phase):

1. A reservoir with N neurons is initialised.

2. The reservoir is driven by all training signals, and the excitations of the neurons of the network (i.e. the response of the reservoir to the signal) are recorded. For each labelled instance s_j of signal/pattern class j in the training set, let the excitations be collected in an $N \times T$ shaped matrix X^j, where each column $x(t)$ is the reservoir's response at time step t. Let K^j be the number of instances of pattern j present in the training data, and the k^{th} example of the signal be s_j^k (which itself is a column vector of size T).

3. The excitations, as well as the input signal, are concatenated into a column vector z_j^k of dimension $dim(z_j^k) = T * (N + M)$, where $k \in \{1, ..., K^j\}$ is an index of the sample. z_j^k is of the form

   $$z_j^k = [x(1); s_j^k(1); ...; x(T); s_j^k(T)] \tag{1.5}$$

   Concatenate all z_j^k into a $dim(z_j^k) \times K^j$ matrix called Z_j.

4. Next, compute a correlation matrix as:

   $$R_j = Z_j Z_j' / K_j \tag{1.6}$$

5. Finally, calculate a conceptor C_j using Eq. 1.3 with some aperture α. Note that both R_j and C_j are shaped $dim(z_j^k) \times dim(z_j^k)$.

Note that these conceptor matrices are *not* the same as one computed purely from the excitation values as described in Section 1.2.2, and have different shapes. The conceptor matrices used for classification described in this section are higher dimensional, and contain temporal information about the neuronal excitations and the pattern itself.

Testing Phase:

1. With the same reservoir as in the training phase, run a candidate signal of shape $M \times T$ through it, and record the reservoir's response.

2. Concatenate the excitations and signal to create a vector z, in the same way as z_j^k was computed in the training phase.

3. For each pattern class j, calculate a *positive evidence quantity*:

   $$h_j = z' C_j z \tag{1.7}$$

   Informally, this value calculates how closely the reverberations of z resemble that represented by C_j for a pattern j. Let there be n total conceptors learned corresponding to n pattern classes. Thus, we can classify an input signal as pattern $\hat{j}$ by choosing the pattern whose conceptor allows for the highest evidence quantity, i.e.

   $$\hat{j} = \underset{i \in \{1,...,n\}}{\text{argmax}}\ h_i = \underset{i \in \{1,...,n\}}{\text{argmax}}\ z' C_i z \tag{1.8}$$

Informally, the evidence value $z'Cz$ can be seen as a measure of how 'closely' the vector z fits into the linear space characterised by C – the higher the value, the better the fit. Thus, the goal of the classification paradigm above is to find the conceptor C_p (where p is a phoneme class, $p \in \mathcal{P}$) which characterises the trajectory of z best. Additionally, searching for a good aperture for calculating conceptors in the training phase is important in order to achieve optimal classification performance over all classes.

1.4 Conceptor Logic

Since a conceptor can be seen as a representation of a signal class with respect to the neuronal response that it creates in a reservoir, then an advantageous attribute that conceptors carry is the ability to be combined in a symbolic way. Specifically, Jaeger (2014) defined a set of Boolean logic-analogous operators that can be used to tackle tasks such as pattern regeneration, time-series predictions, hierarchical classifications etc. All operators operate on conceptors and result in a conceptor.

Though Jaeger (2014) defines a set of equivalent Boolean operator calculations, the definitions of these operators below follow from Mossakowski et al. (2018) and were implemented in code during this project.

- **NOT ($\neg$)**: For a conceptor A, the logical negation is defined as:

$$\neg A := I - A \tag{1.9}$$

  where I is the identity matrix of the same shape as A. Intuitively, $\neg A$ can be seen to be the conceptor that encompasses the linear subspace of neuronal activation that A does *not* occupy. This can be seen as analogous to the set-theoretical operator A^c (*complement*) on a set A, given a universal set U (i.e. $A^c = U \setminus A$).

- **OR ($\vee$)**: For conceptors A and B, the logical disjunction is defined as:

$$A \vee B := C(R(A,1) + R(B,1), 1) \tag{1.10}$$

  where C is the conceptor defined in Eq. 1.3, and R is the correlation matrix from which a given conceptor is computed, defined in Eq. 1.4. Intuitively, $A \vee B$ can be seen to be the conceptor that encompasses the linear subspace of activations that *either* A or B encompass, or as a linear combination of both. It can also be seen as the conceptor that is computed using the *all* the data that was used in the creation of either of the conceptors A or B. With this interpretation, it can be seen as an analogue to the set-theoretical operator $A \cup B$ (*union*) on two sets A and B.

- **AND ($\wedge$)**: For conceptors A and B, the logical conjunction is defined using de Morgan's rule and the definition of disjunction in Eq. 1.10:

$$A \wedge B := \neg(\neg A \vee \neg B) \tag{1.11}$$

  Analogous to the other operators above, $A \wedge B$ can be seen to be the conceptor that encompasses the linear subspace of activations that *both* A and B encompass. With this interpretation, it can be seen as an analogue to the set-theoretical operator $A \cap B$ (*intersection*) on two sets A and B.

For a set of conceptor matrices $\mathbf{C} = \{C_1, ..., C_n\}$, we write their disjunction as:

$$\bigvee_{i=1...n} C_i := C_1 \vee ... \vee C_n \tag{1.12}$$

Likewise, we may do the same for their conjunction as:

$$\bigwedge_{i=1...n} C_i := C_1 \wedge ... \wedge C_n \tag{1.13}$$

1.5 Fuzzy Sets and Fuzzy Logic: A Quick Introduction

Boolean logic is a 'crisp', or two-valued logic – all statements are either true or false. In comparison, fuzzy logic is a system of formal logic where the truth value of sentences may lie in between the interval $[0, 1]$, where 0 is false and 1 is true. The values in between true and false correspond to degrees/gradations of truth – a statement may be true *and* false at the same time to complementary levels (Zadeh, 1975). The benefit of fuzzy logic is that it provides a method to quantify subjectivity, uncertainty, vagueness and ambiguity in line with humans' use of language and reasoning. It is not always the case that a sentence or series of arguments in English necessarily is valued as true or false.

For example, the sentence:

Jakub is quite tall.

is *neither* true nor false on its own, and only makes sense to humans in the context of a particular situation or reference. If we strictly define the limits of the meanings of the words, e.g. defining 'tall' to be a property of a person who has a height greater than some measurable threshold value, the sentence will have a *crisp* (true/false) truth-valuation – however, humans do not generally reason this way, we allow for some *fuzziness* in our words (Zadeh, 1975).

Fuzzy logic was a natural extension of what is known as fuzzy *sets* (Zadeh, 1996). It considers the fact that objects in the physical world belong to classes whose criteria for membership (i.e. whether they belong to the set or not) are generally not precisely defined. Thus, a fuzzy set is defined to be a set that has a continuous, graded membership. This is opposed to 'crisp' sets, corresponding to the more common variety of mathematical sets (such as with Zermelo–Fraenkel set theory), where objects are either members or not members of a set. However, for some classes, especially those that are described by every-day linguistic terms, it is natural to think of them as having *graded memberships*. For example, a crisp version of the following set:

The set of tall men.

may or may not include Jakub from the previous example, but depending on context, he may be regarded as belonging to the fuzzy version of that set as a *function* of his height (for example, the taller he is, the more of a member he is of the set of tall men). Typically, memberships to fuzzy sets are mathematically represented as a function from a n-dimensional real-valued space of characteristics of an object to the real interval $[0, 1]$, where 0 represents not being a member of the set whatsoever, and 1 representing definitely being part of such a set. These are called 'membership functions' and are denoted $\mu_S(x)$, which denotes the membership value of an object x in set S (answering the question *What is the S-ness of x?* or *How S-like is x?*). Thus, fuzzy set theory provides a framework for dealing with ambiguous linguistic classes and terms that one may encounter and use in everyday language.

1.5.1 Neuro-fuzzy systems

Uncertainty in DL has often been quantified in terms of probability. Though fuzzy logic and set theory was recognised early on as a means of mathematically dealing with fuzzy concepts to help in quantifying uncertainty in tasks such as pattern recognition (Bellman et al., 1966) and control systems (Kosko & Burgess, 1998), it has had arguably limited success in recent years due to the practical success of (deep) machine learning. Notably, some attempts have been made to merge neural networks and fuzziness, in a field called *neuro-fuzzy* systems (for an overview, see Abraham, 2005). Bottom-up, data-driven learning systems such as neural networks are more robust to noise than rigid rules, however, rule-based or logical systems benefit from being structured and generally interpretable – neural networks have replaced these systems with subsymbolic, non-linear transformations, which are far less interpretable. Since this project, in part, aims to show potential benefits of fuzziness with reservoir computing and conceptors, it may be fair to call the approach presented in Section 3.4.3 a neuro-fuzzy system.

1.5.2 Fuzziness versus Probability

Fuzzy logic, as mentioned before, provides a mathematical basis for quantifying ambiguity or uncertainty. One may argue that probability does too. Both systems are similar in that uncertainty is described with a real-valued number between 0 and 1. However, a notable difference between the fuzzy and probabilistic interpretations of the same ambiguous situation is a conceptual one – fuzziness does not assume that there is one true answer, whereas probability does (Kosko & Burgess, 1998).

Fuzziness quantifies the question *How much...?*, whereas probability answers *Whether...?*. *Whether* an event occurs is 'random', is interpreted as probabilistic, and may be simulated by drawing from an appropriate probability distribution. The probability of a fair coin landing heads-up and that of it landing tails-up are required to sum to 1. *If* this coin is flipped, it may be said that it landing heads-up has a (frequentest) probability of 0.5, but the 'heads-up-ness' of the coin cannot be said to be 0.5. Additionally, the sum of an object's membership to multiple sets do not have this requirement

Contrariwise, fuzziness asks *to what degree* an event occurs, not whether or it has. A hybrid fruit between an orange and a lemon may be belong to both the fuzzy sets 'orange' and 'lemon' to differing degrees, since it may possess traits of both, but to say it has a probability of being one or the other is less meaningful. The question *What is the likelihood of this being a lemon?* has a completely different meaning to *How much like a lemon is this?*. The fruit's membership values representing 'orange-ness' and 'lemon-ness' are somewhat independent of other membership values such as 'grapefuit-ness' or 'tangerine-ness' (since it may also posses comparable traits of these other classes such as sweetness or being in the citrus family). The total of the membership values a fruit may possess is not required to sum to unity.

1.6 *Fuzzy* Conceptor Logic

Though Jaeger (2014) describes several aspects of combining conceptors through the logic system he calls 'Boolean' (described in section 1.4), Mossakowski et al. (2018) suggest that conceptor logic is better interpreted as *fuzzy*. Particularly, this is motivated by the fact that conceptors are computed based on generally noisy data – this brings with it a notion of uncertainty of the identity of the signal. Informally, they propose that conceptors may be treated analogously to fuzzy sets. Importantly with respect to this project, they also analyse the relationship of a signal to a conceptor within the classical conceptor classification paradigm, and propose to have a fuzzy membership function based on the evidence value described in Eq. 1.7 as:

$$\mu_C(z) := \frac{1}{D} z' C z \tag{1.14}$$

where C is a conceptor associated to a certain signal class, z is the column vector of concatenated reservoir excitations and pattern as described in Eq. 1.5, and $D = dim(z) = dim(C)$ is the dimension of z and the conceptor C. Thus, this can be understood as the evidence value $z'Cz$ normalized with respect to the number of dimensions of C – informally, this answers how much z conforms to the subspace of C averaged over all directions. Mossakowski et al. showed that $\mu_C(z)$ has the properties of fuzzy membership with respect to the signal class, namely, that $\mu_C(z) \in [0, 1]$.

1.7 Aperture adaptation

Recall that computing a conceptor relies on an an 'aperture' value α. With the geometric intuition of a conceptor corresponding to a linear subspace of neuronal activations, α can be seen as a regulating parameter of the balance between how large the lower singular values of the correlation matrix are taken into account compared to the higher ones – a larger aperture implies that the axis of the ellipse will be larger for the directions of the lower-powered singular values, or that more of the lower singular values are taken into account when computing the conceptor-representation of the trajectory. It can also be interpreted as a scaling factor of the reservoir data, or scaling the signal energy by α^2. If α is set too low, the resultant conceptor may not adequately or fully encompass the space of activations of the underlying signal. However, if α is set too high, the conceptor may begin to encompass significant space in the singular directions that may not represent the signal itself – i.e. the conceptor may begin to encompass noise.

A conceptor may be recomputed with a different aperture relative to the one it was trained with using an operation called *aperture adaptation.* This operation on conceptors is defined by Jaeger (2014) as:

$$\varphi(C, \gamma) = C(C + \gamma^{-2}(I - C))^{-1} \tag{1.15}$$

where I is the identity matrix the same shape as C, and $\gamma \in [0, \infty]$ is an *aperture adaptation factor.* This operation scales the aperture of the conceptor by a factor of γ.

1.7.1 Aperture-adapted Disjunction and Conjunction

Let us assume we have an set of conceptor matrices $\mathbf{C} = \{C_1, ..., C_n\}$ whose apertures we wish to adapt respectively to corresponding factors $\Gamma = \{\gamma_1, ..., \gamma_n\}$, and after which, whose disjunction is taken. This process will be termed *aperture-adapted disjunction* and denoted by:

$$\bigvee_{i=1...n} \varphi(C_i, \gamma_i) := \varphi(C_1, \gamma_1) \vee ... \vee \varphi(C_n, \gamma_n) \tag{1.16}$$

For completeness, we may also define an *aperture-adapted conjunction* operator, as the name implies, as the conjunction of the respectively aperture-adapted conceptors above as:

$$\bigwedge_{i=1...n} \varphi(C_i, \gamma_i) := \varphi(C_1, \gamma_1) \wedge ... \wedge \varphi(C_n, \gamma_n) \tag{1.17}$$

1.8 'Optimal' Aperture Computation

Let us assume that for a conceptor C, there exists a corresponding aperture α_{opt} that optimally represents the data corresponding to the signal class it were trained on. Though there does not yet exist a rigorous definition of a 'goodness' or 'optimality' of this type (Jaeger, 2014), we can define one with respect to the task.

For classical conceptor classification, Jaeger (2014) defines a '∇-rule', which monitors the gradient of the squared Frobenius norm of a conceptor originally trained with $\alpha = 1$ with respect to the logarithm of $\log(\gamma)$, which is an aperture adaptation factor as per Eq.1.15. We can then choose the value of γ that maximises the function in Eq 1.18.

$$\nabla(\gamma) = \frac{d}{d\log(\gamma)}||\varphi(C, \gamma)||^2 \tag{1.18}$$

This may be computed for each conceptor by sweeping through a large range of γ, computing the squared Frobenius norm, and approximating the derivative. Let the sequence of γ be enumerated by an index n. Now, define the difference between two sequential log-transformed γ values as:

$$\Delta\gamma(n) = \log(\gamma(n+1)) - \log(\gamma(n)) \tag{1.19}$$

Likewise, we can define the difference in sequential Frobenius norms as:

$$\Delta F(n) = ||\varphi(C, \gamma(n+1))||^2 - ||\varphi(C, \gamma(n))||^2 \tag{1.20}$$

Thus, based on the definition of the (right-hand) derivative:

$$\frac{d}{dx}f(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h} \tag{1.21}$$

with $f(x) := F(n)$, $h := \Delta\gamma(n)$, we get an approximation of the derivative, since $\Delta\gamma(n)$ does not necessarily tend to zero.

$$\nabla(\gamma(n)) \approx \frac{\Delta F(n)}{\Delta\gamma(n)} \tag{1.22}$$

For each conceptor, we choose the γ that is maximal as γ_{opt}, and adapt the conceptor to $\varphi(C, \gamma_{opt})$. The semantics of this rule can be found in Jaeger (2014) and is outside the scope of this thesis. Other optimal aperture calculations such as searching for the set of α that maximise performance of some kind are also possible. However, this project uses the ∇-rule for finding the 'optimal' apertures for all conceptors, since this criterion allowed Jaeger (2014) to achieve high accuracy in a separate classification task.

1.9 Aperture Adaptation as a Function of Uncertainty

We can tentatively claim that this optimal aperture allows the conceptor to model the signal with high 'certainty'. When we are certain a signal belongs to a class, the linear excitation subspace represented by the conceptor is well adjusted to the trajectory of the signal. If we adjust the apertures of the conceptors with a scaling factor $\gamma < 1$, this would result in allowing for an amount of *uncertainty* in the evaluation of a test example's excitation space encapsulated by the conceptor. Note that this interpretation is one of many possible interpretations. The exact shape of γ as a function of uncertainty is not known, and was not investigated during this project due to time constraints.

For a conceptor C, let the optimal aperture be α_{opt}, and let C be computed using α_{opt} according to Eq. 1.3 or suitably aperture-adapted to have α_{opt}. Let there be some uncertainty quantifier $\Psi(s)$ that arises with respect to a symbol s, contextualised by the task. We wish to adapt the aperture of C by some scaling factor γ (using Eq. 1.15) to represent this. Thus, we may expect there to exist a function $\gamma(\Psi(s))$, which represents an aperture adaptation based on some uncertainty function $\Psi(s)$.

According to the previous interpretation of uncertainty with respect to the scaling of the aperture of the conceptor C away from α_{opt}, we may intuitively expect $\gamma(\Psi(s))$ to be bound in the range $[0, 1]$. If our expectation of a symbol with respect to our task is totally certain, e.g. we expect the next classification to be a particular symbol class, then $\gamma(\Psi(s)) = 1$, and the conceptor is left optimally adapted. If we are certain that our expectation is *not* the that the conceptor represents, then $\gamma(\Psi(s)) = 0$, and when aperture-adapted, the conceptor becomes a zero matrix. If we are uncertain about whether we expect this symbol or not, $\gamma(\Psi(s)) \in (0, 1)$. This mathematical notion is mirrored in other uncertainty quantification paradigms such as probability and fuzziness.

1.10 *Biased* Conceptor Classification Paradigm

This section describes how the panorama of topics covered in sections 1.1 to 1.8 are combined to form a novel and cohesive procedure of biasing a reservoir's neuronal activations, used to augment classical conceptor classification.

The three biased classification schemes presented all follow the same basic principle: use contextual information about a symbolic prediction $\hat{s}_t$ at time t to create a conceptor to change/prepare the reservoir for an prediction $\hat{s}_{t+1}$ at the next time step $t+1$. All three assume that the symbol class of the next time-step is stochastically determined by the current one.

The process constituting 'biasing' is as follows: given a ('biasing') conceptor B, the reservoir excitations with respect to the *previous* time-steps can be transformed or biased in order to regulate the excitations in the current time step. This can be done by modifying the reservoir's state update rule (in Eq. 1.1) to:

$$x(t+1) = B \tanh(Wx(t) + W^{in}s(t+1) + b) \quad (1.23)$$

In this way, the excitations within the reservoir are transformed or 're-shaped' towards the linear subspace that is determined by B. We then use the classical conceptor classification paradigm to make class predictions using Eq 1.23 to drive the reservoir. If the conceptor B is generated from global or top-down data, then it may be claimed that top-down information influences the bottom-up predictions of the system via biasing the state updates of the reservoir.

In the context of phoneme transcription, where a phoneme is defined as a symbol, this idea should help the system disambiguate similar phonemes, since, although the phonemes may be similar in terms of raw audio data, they do not often occur in the same phonemic contexts – for example, given the previous phoneme /ae/, we do not expect to hear the phoneme /ng/ as much as we expect /n/, thus, we interpret the phoneme as /n/ (i.e. the phoneme pair '/ae n/', e.g. in the word "hand", may have more occurrences in our training data more often then '/ae ng/').

The three systems generate these biasing conceptors in different ways, inspired by different interpretations of human reasoning. The first is termed *symbolic*, and is based on reasoning that uses 'crisp' notions of symbols. The second, termed *probabilistic*, is based on symbols whose uncertainty is quantified in terms of probability. The third one, termed *fuzzy*, is based on symbols whose uncertainty is quantified in terms of fuzzy membership. The bias conceptors B are created by an aperture-adapted disjunction of conceptors that correspond to the next symbolic phonemes predicted, using probability and fuzziness to guide the aperture adaptation. The details for generating these bias conceptors can be found in section 3.4.

2 Task & Data Set

Since this project aims to be an exploratory starting point in using context to disambiguate similar input signals, the toy task of phoneme classification in English was used. The TIMIT data set (Garofolo et al., 1993) was used for this task, as it contains the audio recordings of a large variety of (American) English speakers as well as time-segmented and labelled phonemes. Since phonemes are the most primitive symbolic building blocks of speech, which is often noisy and ambiguous, it lends itself to being an interesting symbolic-subsymbolic integration task.

Large phonemic variation across different spoken accents, as well as frequency variation amongst genders (women usually have higher pitched voices than men) may prove problematic to analysing the behaviour of the system – more variation in the way each phoneme class is represented sub-symbolically (i.e. on the data level) would require a larger reservoir and thus larger conceptors to suitably classify them apart, which may become computationally expensive. Thus, only sentences spoken by male speakers of the New England dialect region was used (denoted 'DR1' in the data set).

The training set consists of 24 speakers speaking a total of 240 sentences, and the test set consisted of 7 speakers, speaking a total of 70 sentences. The raw audio data has a sampling rate of 16kHz, and was in monaural format (one signal channel over time). In total, there are 61 labels defined in the training data including phonemes and silence.

Let $\mathcal{P}$ be defined as the finite set of phoneme classes that exist in the given data set. The ordering and indexing of phonemes is arbitrary as long as it is consistent, but was done alphabetically by their labels. The list of all phonemes represented in this data and the distribution of their sample occurrences can be seen in Table A.1.

The training and testing audio data is windowed over time, and a set of concurrent, overlapping windows in the (mel-)frequency-domain are input for classification. One sequence of windows in this format will be called an '*audio-snippet*'.

Thus, informally, the final task is defined as: given an input audio signal, classify each audio-snippet as being part of a phoneme from a set of known phoneme classes.

2.1 Pre-processing

The raw audio data was first pre-processed and translated from a time-domain to a frequency-domain representation. This was done by taking a mel-frequency cepstrum coefficient (MFCC) representation for short audio windows (Lyons et al., 2020). This is often done with raw audio data in tasks such as speaker recognition (Ganchev et al., 2005) and automatic phoneme transcription (Graves & Schmidhuber, 2005). This change of representation is useful, since some speech features important to classification, such as formants, are better represented in the frequency-domain than the time-domain. Additionally, it is known to reduce the effects of noise (Huang et al., 2001).

Using the Python library published by Lyons et al. (2020), 15 MFCCs (coefficients) with 26 filter-bank channels were taken for each window in a sequence of overlapping windows of length 15ms and a step size of 7.5ms. Each input sample (i.e. an 'audio-snippet', as named previously) consisted of $T = 5$ concurrent windows (again, each overlapped with the ones adjacent to it). For reference, the mean number of concurrent windows per segmented phoneme in the training and testing data is approximately 10.

Thus, the n^{th} audio-snippet can be represented as a matrix $U(n)$ of shape $M \times T = 15 \times 5$ where M is the number of coefficients. The columns of $U(n)$ are denoted $s(t)$, which represents individual time steps of the transformed audio-data. This was done for the audio data for every sentence in the training and test sets.

Additionally, if an audio-snippet sample fell *wholly* within the bounds of a segmented phoneme, it was labelled an instance of that phoneme. If an audio-snippet consisted of information from two or more phonemes, i.e. it fell across phoneme boundaries, it was discarded. In total, this resulted in 79553 input samples in the training set, and 21326 in the test set. The distribution of phonemes represented by these samples is quite skewed, as can be seen visually for the training and test sets in Figures A.1 and A.2 respectively.

3 Methods

This section discusses the specific steps taken to run the phoneme classification task with the classical and biased paradigms (cf. section 1.10). This was implemented in Python 3.7.3, and the code can be found at `https://github.com/satchitchatterji/ContextualConceptors`.

3.1 Reservoir Initialisation

A single reservoir of $N = 50$ neurons was initialised. This was achieved by creating an $N \times N$ matrix W of 70% sparsity with entries that were sampled from a normal distribution $\mathcal{N}(0, 1)$, representing the weights of the recurrent connections between the neurons. A bias vector b was created by sampling from a normal distribution $\mathcal{N}(0, 0.1)$. Finally, an input weight matrix W^{in} was created of shape $N \times M = 50 \times 15$, whose entries were sampled from a normal distribution $\mathcal{N}(0, 1)$.

The reservoir weight matrix was then altered in order to guarantee that it had the 'echo state property': any initial state of a reservoir is forgotten after a given period such that the excitations of the reservoir are just a function of the input signal. This has been shown to be an important condition for recurrent neural networks in learning and control tasks (Jaeger, 2014). The echo state property was guaranteed using the procedure outlined in Yildiz et al. (2012), as described here:

1. First, a random weight matrix with non-negative entries is needed. Thus, each of the entries of the previously initialised weight matrix were replaced with their absolute-values.

2. W was scaled such that its spectral radius $\rho(W)$ was less than 1. Here, the spectral radius refers to the largest absolute eigenvalue of W. Thus, W can be scaled as $W \leftarrow W/(\rho(W)+\epsilon)$, for an arbitrary small positive value ϵ (here, ϵ was chosen to be 0.005). If $\epsilon = 0$, then $\rho(W) = 1$. The value ϵ ensures that the new spectral radius $\rho(W)$ is less than 1.

3. The signs of a certain proportion of entries were flipped, i.e. a random sample of the entries of W were made to be negative, chosen here to be 50%.

The weights and biases of this reservoir will no longer be changed.

3.2 Training: Learning Conceptors

For each phoneme class $p \in \mathcal{P}$, two conceptors of different purposes were created which were called C^p_{class} and C^p_{bias} respectively, with audio-snippets of class p used to drive the reservoir. All neurons in the reservoir had an initial excitation state of zero, and this is discarded when creating the state collection matrices.

These two types of conceptors were used in order to facilitate both classification and biasing. C^p_{class} is a high-dimensional representation of both the excitations of the neurons and the signal itself, and includes temporal information (since they are created by flattening the excitations over time into a vector). These are thus used by Jaeger (2014) to define the classical conceptor classification paradigm – a method which is extended in this project. Since this does not directly represent the subspace of neuronal excitations with respect to the total space of the reservoir, a second conceptor type C^p_{bias} is used to transform or bias the space of excitations of the reservoir as per Eq 1.23, and is the same shape as the reservoir's weight matrix.

The first type of conceptor, C^p_{class}, created to enable the classical conceptor classification scheme as described in section 1.3, were square matrices of dimension $T*(N+M) = 5*(50+15) = 325$ (i.e. the shape of the matrix is 325×325), where T is the number of time steps of the input audio-snippet, M is the dimension of the input vector (i.e. the number of MFCCs of each time step in the audio-snippet) and N is the number of neurons in the reservoir. Let the set of all conceptors of this kind be $\mathbf{C}^{\mathcal{P}}_{class}$, indexed to correspond with the order of $\mathcal{P}$.

The second type of conceptor, C^p_{bias}, created to enable the biasing of the reservoir, were computed according to the procedure described in section 1.2.2, and represents the linear subspace that encapsulates the trajectory of the excitations of the neurons of the reservoir when driven by examples of phoneme p. The shape of this conceptor is $N \times N = 50 \times 50$. Let the set of all conceptors of this kind be $\mathbf{C}^{\mathcal{P}}_{bias}$, again ordered to correspond with the order of $\mathcal{P}$.

The apertures that C^p_{class} and C^p_{bias} for all phonemes $p \in \mathcal{P}$ were set to their respective 'optimal' apertures based on the ∇-rule defined in 1.8, though other methods may be used. The $\nabla(\gamma)$ functions for a sample of conceptors is given in Figures 3.1a and 3.1b for visual reference on how α_{opt} was computing using this procedure.

3.3 Testing I: Unbiased Classification

An *unbiased* classification refers to the classical conceptor classification paradigm as described in section 1.3. To summarise the procedure, an unseen audio-snippet is used to drive the reservoir, and the excitations collected. A column vector z is created which consists of concatenated excitations of the reservoir and the signal itself.

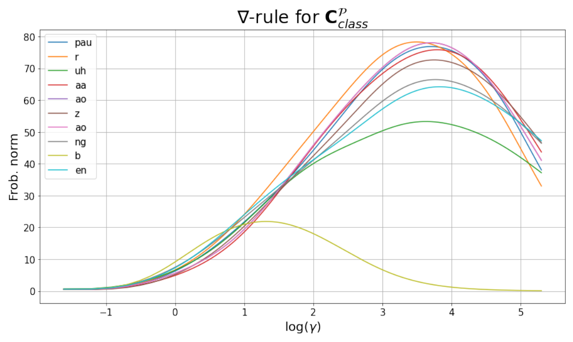
For each phoneme class $p \in \mathcal{P}$, the corresponding conceptor C^p_{class} is used to calculate an evidence value for the signal as

$$h = z' C^p_{class} z \qquad (3.1)$$

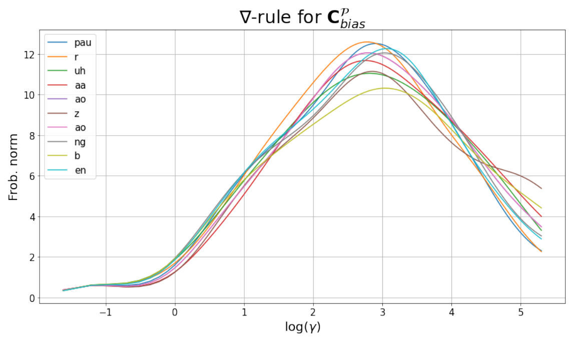
The audio-snippet then is classified into the phoneme class $\hat{p}$ with the highest corresponding evidence value.

3.4 Testing II: Biased Classification

The three systems described below generate these biasing conceptors in different ways, inspired by different interpretations of human reasoning. The *symbolic* and *probabilistic* biasing methods rely on



(a) **Frobenius norm over $\log(\gamma)$ for $\mathbf{C}^{\mathcal{P}}_{class}$.**



(b) **Frobenius norm over $\log(\gamma)$ for $\mathbf{C}^{\mathcal{P}}_{bias}$.**

Figure 3.1: Visualisation of the ∇-rule for computing optimal apertures of the $\mathbf{C}^{\mathcal{P}}_{class}$ (*left*) and $\mathbf{C}^{\mathcal{P}}_{bias}$ (*right*) conceptors for a sample of 10 random phonemes. This was done for all conceptors, and γ_{opt} is chosen at the point of maximal Frobenius norm.

a Markov matrix consisting of conditional probabilities of phonemes occurring after one another in the training dataset. This is described in appendix A.2. Additionally, the *fuzzy* method relies on predicting fuzzy phoneme membership values of the next audio-snippet given the current one. This was done using a multi-layer perceptron and is described in appendix A.3.

3.4.1 Symbolic Biasing

This method generates a biasing conceptor B by looking at whether a particular symbol class may be expected at the next time step at all – that is, bias based on all possibilities seen in the training data. Let a sequence of symbols (s_i, s_j) be called a bigram (as it is often called in natural language processing when using words as symbols instead). Thus, symbolic biasing here is based on whether a phonemic bigram (p_i, p_j) exists for a set of phonemes $p_j \in \mathcal{P}$ in the training data set.

Thus, if the system predicts phoneme p_i at time t, it looks for possible sequences or bigrams of the form (p_i, p_j). This amounts to looking at the row of the Markov matrix that corresponds to the latest prediction and looking for symbols whose conditional probability is greater than zero. This process was termed 'symbolic' in order to encapsulate the concept of 'crisp' symbolic human reasoning that can be modelled by classical logic, i.e. the symbols have no ambiguity or certainty associated with them.

For the current audio-snippet at time t, let $\mathbf{C_{next}} \subset \mathbf{C}^{\mathcal{P}}_{bias}$ be the set of bias conceptors corresponding to the next *possible* phoneme classes at time $t + 1$. The biasing conceptor B is created as the disjunction of all of these:

$$B = \bigvee \mathbf{C_{next}} \tag{3.2}$$

The above bias can be interpreted as: *Given our prediction of phoneme p_t, the next phoneme is expected to be p_1 or p_2 or ... or p_n since the phoneme pair (p_t, p_i) exists in the training set for $i = 1, ..., n$.*

For the sake of computational equivalence with respect to the other two biasing methods, this can also be written out as an aperture-adapted disjunction. The aperture adaptation factors γ_i for each conceptor $C^{p_i}_{bias} \in \mathbf{C}^{\mathcal{P}}_{bias}$ are either 1 or 0 based on whether the bias conceptor corresponds to a phoneme class that deemed possible in the next time step. This can be written concisely as:

$$\gamma_i = \begin{cases} 1 & \text{if } C^{p_i}_{bias} \in \mathbf{C_{next}} \\ 0 & \text{if } C^{p_i}_{bias} \notin \mathbf{C_{next}} \end{cases} \tag{3.3}$$

This ensures that the conceptors in $\mathbf{C_{next}}$ remain at their optimal aperture, and all others are reduced to the zero-matrix. Thus our aperture-adapted disjunction is:

$$B = \bigvee_{i=1,...,61} \varphi(C^{p_i}_{bias}, \gamma_i) \tag{3.4}$$

This bias can now be interpreted as: *Given our prediction of phoneme p_t, the next phoneme is certain* *to be p_1 or p_2 or ... or p_n and not any others since the phoneme bigram (p_t, p_i) exists in the training set for $i = 1, ..., n$.*

3.4.2 Probabilistic Biasing

This method generates a biasing conceptor B by looking at the probability of a particular symbol class being present at the next time step. For this, we read the row of the Markov matrix (see appendix A.2) that corresponds to the classification p_t of audio-snippet phoneme at time t. Let this row be represented as a row vector of conditional probabilities $P(p_i|p_t)$ for all $p_i \in \mathcal{P}$. We can treat these probabilities as the function of uncertainty. Thus, we define an uncertainty quantifier around p_i with respect to p_t to be:

$$\Psi_i(p_t) := P(p_i|p_t) \tag{3.5}$$

Let the aperture adaptations be $\gamma_i(\Psi_i(p_t))$ for phoneme-corresponding indices $i \in \{1, ..., 61\}$. We may define $\gamma(x)$ to be the identity function for simplicity. Thus for each phoneme class i:

$$\gamma_i(\Psi_i(p_t)) := P(p_i|p_t) \tag{3.6}$$

This follows the conditions laid out for $\gamma(\Psi(s))$ in section 1.9 – namely, that it is bound in $[0, 1]$. Additionally, it allows us to understand the aperture adaptation directly as a correlate for probabilistic uncertainty. Finally our bias conceptor can be calculated to be:

$$B = \bigvee_{i=1,...,61} \varphi\,(C^{p_i}_{bias}, P(p_i|p_t)) \tag{3.7}$$

We may use a disjunction of all conceptors, correspondingly aperture-adapted to their uncertainties. The above bias can be interpreted as: *Given our prediction of phoneme p_t, the next phoneme is expected to be p_1 or p_2 or ... or p_n but with some conditional probabilities $P(p_1|p_t)$, $P(p_2|p_t)$, ..., $P(p_n|p_t)$ learned from the training set.*

3.4.3 Fuzzy Biasing

This method is similar to the one described previously with probabilities, however, instead of calculating probabilities from one time step to another, we use the fuzzy membership values based on the evidence values of the current time step. For the audio-snippet at time t, 61 evidence values are calculated as per Eq. 1.7 (one for each conceptor in $C^{\mathcal{P}}_{class}$), from which we can calculate 61 fuzzy membership values as per Eq 1.14, i.e. by normalising it with respect to the number of dimensions of each conceptor in $C^{\mathcal{P}}_{class}$ (i.e. divide the evidence value by 325).

Thus, a phoneme such as /m/ can have some degree of /n/-ness, and vice versa. This holds for all pairs of phonemes – every phoneme instance may hold some graded membership in every phoneme class. Thus, for now, we ignore the there may exist one single "true" phoneme, and allow for gradation in between.

From the current set of 61 membership values, we aim to predict the next set in order to define the bias conceptor. However, no suitable algorithm was found in previous literature, thus, for now, we assume there exists a function $\mathcal{F} : \mathbb{R}^{61} \to \mathbb{R}^{61}$ that can predict multi-class fuzzy membership values, given fuzzy membership values as an input. Under this assumption, a multi-layered perceptron (MLP), whose details can be found in appendix A.3, was created to model this function. It was trained to predict the fuzzy membership values of the next phoneme, given the current one. Let the current set of membership values of the audio snippet at time t be $\mathcal{M}(t)$. This MLP is notated and treated as a function called $\mathcal{F} : \mathbb{R}^{61} \to \mathbb{R}^{61}$, such that $\mathcal{F}(\mathcal{M}(t)) = \mathcal{M}(t+1)$.

Just like with probabilities, we can treat these fuzzy predictions as a function of uncertainty Ψ – however, this is now defined as a function of membership functions over time. let $\mu_i(t+1)$ be the predicted fuzzy membership value for the fuzzy class p_i at time $t+1$ (i.e. $\mu_i(t+1) \in \mathcal{M}(t+1)$). Thus, the uncertainties around phoneme class i for the future is:

$$\Psi_i(\mu_i(t)) := \mu_i(t+1) \tag{3.8}$$

We again define $\gamma(x)$ to be the identity function. Thus for each phoneme class i, the aperture adaptations are:

$$\gamma_i(\Psi_i(\mu_i(t))) := \mu_i(t+1) \tag{3.9}$$

This also follows the conditions laid out for $\gamma(\Psi(s))$ in section 1.9 – namely, that it is bound in $[0, 1]$. This corresponds with the fuzzy intuition of membership – if an object is fully in a fuzzy set, their membership value approaches 1. When it is totally not a member, the membership value approaches zero. Thus, we may use a disjunction of all conceptors, correspondingly aperture-adapted to their fuzzy uncertainties. Finally our bias conceptor can be calculated to be:

$$B = \bigvee_{i=1,\ldots,61} \varphi\left(C^{p_i}_{bias}, \mu_i(t+1)\right) \tag{3.10}$$

At first glance, this equation looks similar to Eq. 3.7 – however, the underlying uncertainty quantification philosophies and valuations are quite different, specifically in the estimation of Ψ and the aperture adaptation factors $\gamma_i(\Psi)$.

The bias can be interpreted as: *Given the current audio-snippet's fuzzy membership values for each phoneme* $p_i \in \mathcal{P}$*, the next phoneme is expected to have a fuzzy membership value in* p_1 *or* p_2 *or ... or* p_n *for all* $p_i \in \mathcal{P}$ *with graded memberships of each phoneme class* $\mathcal{F}(\mathcal{M}(t)) = \mathcal{M}(t+1)$ *learned from the training set.*

This model necessarily does not consider one phoneme to be *true*, rather, that an audio-snippet can sound like several phonemes at once with varying degrees of membership. However, in order to compare this model to the two other biased models and the unbiased one, the highest membership value was considered to be the system's overall classification.

4 Results

The overall differences in accuracy between all four methods (one classical conceptor and three biased conceptor methods) is small but noticeable. This is documented in Table 4.1. We note that the biased methods are better than the unbiased one by around 2.4%. Additionally, since the phoneme /h#/ is regarded as silence and is over-represented in the testing set, the accuracies excluding this phoneme class are shown as well, with the biased methods showing a similar 2% improvement. Thus, biasing seems to improve accuracies overall.

For reference, if phoneme labels were distributed uniformly within the test set, the random baseline accuracy (randomly choosing a phoneme class for each audio-snippet) is $1/|\mathcal{P}| \approx 1.63\%$. However, since we see that it is non-uniform (Figure A.2), an empirical evaluation was conducted by choosing a uniformly random phoneme class for each audio-snippet and calculating the overall accuracy. This was done 1000 times for each audio-snippet. The mean baseline was found to be 1.66% for all classes, and 1.70% without the /h#/.

Method	**Accuracy**	**Without /h#/**
Unbiased	28.2%	21.9%
Symbolic	30.6%	23.5%
Probabilistic	30.6%	23.4%
Fuzzy	30.6%	23.5%

Table 4.1: Overall accuracies of the unbiased and biased conceptor classification methods.

4.1 Phoneme Similarity & Grouping

Since there are 61 phonemes in total and four classification models, analysing all results in their entirety may not be beneficial in understanding the behaviour of the classification systems. In order to gain insight into specific cases, specifically how the biased classification systems act on similar sounding phonemes, subgroups of similar-sounding phonemes were defined.

A first attempt at grouping phonemes are by looking how similar each phoneme class's conceptor is to every other one. This is a measure of the similarity of the response of the reservoir (in a sense, the 'perception') to input examples of phoneme classes. This is done using the conceptor similarity measure defined in Jaeger (2014) and Eq. 4.1 on the bias conceptors $\mathbf{C}^{\mathcal{P}}_{bias}$.

$$\text{sim}^{\alpha}_{i,j} = \frac{||(S^i)^{1/2}(U^i)'(U^j)(S^j)^{1/2}||^2}{||diagS^i||\,||diagS^j||} \tag{4.1}$$

where US^iU' is the single value decomposition of conceptor $C(R^i, \alpha)$ for a phoneme class p_i. The similarities for all conceptors was computed with $\alpha = 1$, and range in $[0, 1]$. The similarities are shown visually in appendix A.4.

Additionally, the phoneme examples were subjectively heard by the author and grouped according to how similar at least one human perceived the signals to be. Since this project is exploratory, it was decided that this was a sufficient enough measure, though a more thorough determination of signal similarity should be done.

In the end, four phonemes groups were identified, given labels (group names) for identification purposes, and are defined in Table 4.2. Though other groups of similar phonemes exist, and these four groups do not contain all 61 phonemes, it provides a reasonable overview of different types of phonemes – vowels, nasals, plosives, and fricatives.

Group Name	Group Members
Open vowel	aa, ae, ah, ao, aw, ax
Nasal	em, en, m, n, ng, nx
Closed plosive	pcl, bcl, dcl, tcl, gcl, kcl
sh-Group	sh, jh, ch, zh

Table 4.2: Groups of similar phonemes and their group names as assigned by the author. For clarity, the forward slashes that usually surround the phoneme are removed. For details, refer to section 4.1.

4.2 Performance Within Groups

To reiterate, the goal of this project was to see if a biasing paradigm using context-related conceptors on reservoir networks would help disambiguate similar phonemes. Thus, we can analyse these in detail by only looking at the classification of audio-snippets which either have the ground-truth label belonging to the a particular group of similar phonemes or were predicted to have such. This essentially elucidates if the model is now able to classify similar phonemes correctly.

Since we wish to focus on the *difference* that biasing brings to the classical conceptor classification paradigm in terms of misclassification of similar phonemes, we may use confusion matrices. Specifically, the confusion matrices that follow show the *difference* in total true positives, false positives, false negatives and true negatives compared to the unbiased model. For an overall improvement, we would expect there to be an increase in the true positives and true negatives, and a decrease in the false positives and false negatives. For compactness, this is called a *difference confusion matrix.* In the description presented in this section of the values within the difference confusion matrices, the means of all three biasing methods have been presented rounded to the nearest whole number, with specific values being available in the figures 4.1-4.4.

4.2.1 Open Vowel Group

Figure 4.1 shows the results of all the biasing methods based on the open vowel group of phonemes. For the true positives, we see a total increase for all three biased models to be 58 audio-snippets more than the unbiased model. Additionally, The true negatives increase substantially too with each model having a mean increase of around 1129 samples. This would imply that the models are now better at both classifying a phoneme correctly, and not classifying it as another phoneme in the group too. We also see a decrease in the number of false positives in each (58 samples), however, the false negatives increase to a mean of around 281 more samples than the unbiased model. Thus overall we see an improvement in three of the four quadrants of the difference confusion matrices.

4.2.2 Nasal Group

Figure 4.2 shows the results of all the biasing methods based on the nasal group of phonemes. We see a similar behaviour to the Open Vowels, but less pronounced in terms of absolute values of change. The true positives and true negatives increase by about 102 and 71 respectively. The false positives decrease by approximately 102, however, the false negatives increase for each model by 168 samples. This also indicates an improvement in three of the four quadrants of the difference confusion matrices.

4.2.3 Closed Plosive Group

Figure 4.3 shows the results of all the biasing methods based on the closed plosive group of phonemes. For this group, we see a general decrease in performance. The true positives decrease by 32 samples. However, the true negatives increase to 193 samples for each – however, the symbolic and fuzzy biasing methods do far better than the probabilistic method in this regard, with a respective increase of 251, 251 and 77 samples each. The false negatives increase by 32 samples each, and the false negatives increasing by around 97 each. This phoneme group shows greater differences between the models in performance than the others. Thus, we only get an improvement in the true negatives.

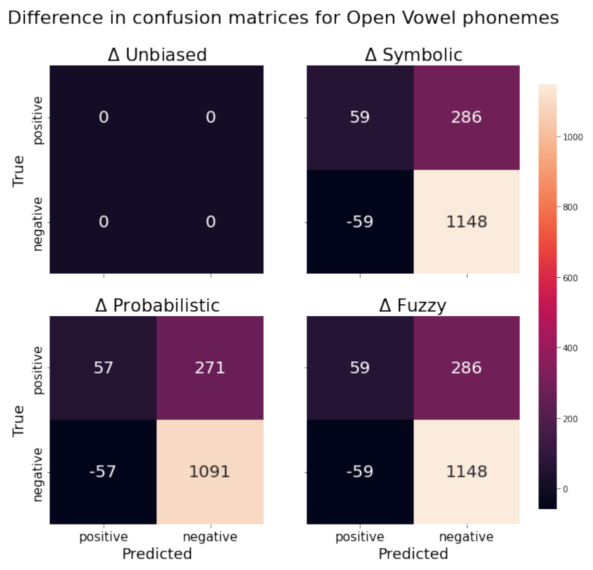


Figure 4.1: Difference in confusion matrices for total classifications of phonemes with respect to the *Open Vowel* group.

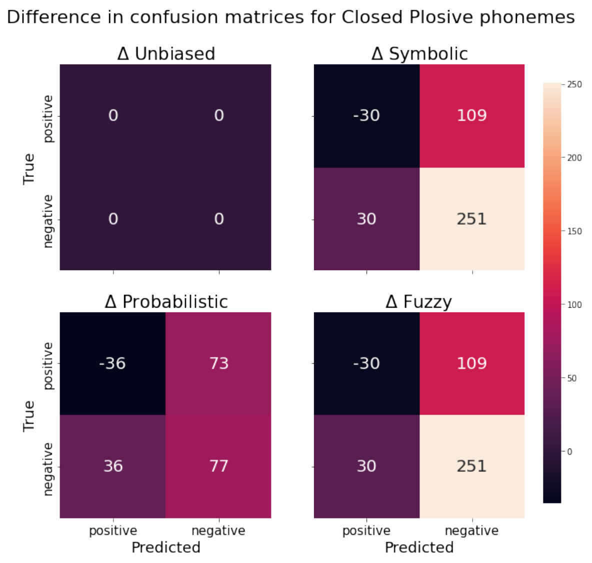


Figure 4.3: Difference in confusion matrices for total classifications of phonemes with respect to the *Closed Plosive* group.

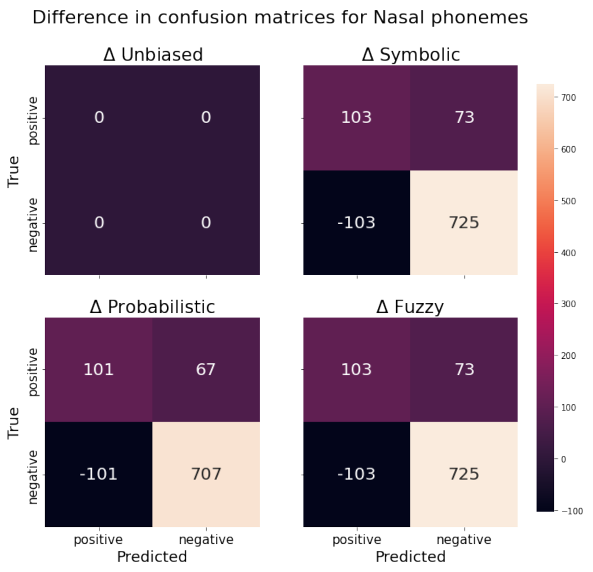


Figure 4.2: Difference in confusion matrices for total classifications of phonemes with respect to the *Nasal* group.

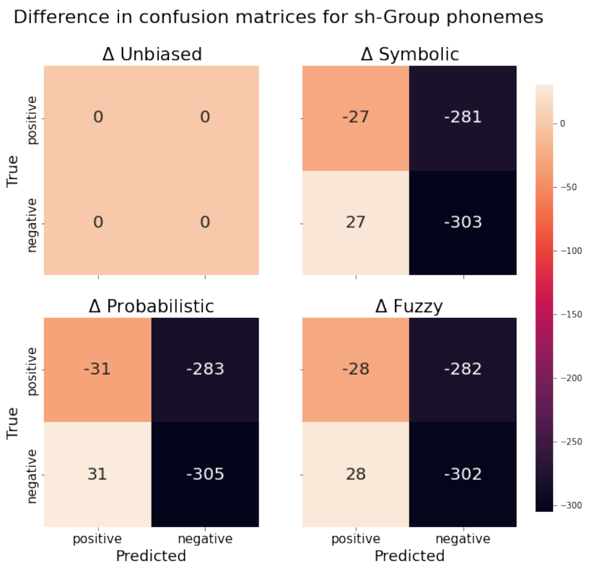


Figure 4.4: Difference in confusion matrices for total classifications of phonemes with respect to the *sh-Group*.

4.2.4 sh-Group Group

Figure 4.4 shows the results of all the biasing methods based on the sh-group of phonemes. For this group, we see a general decrease in performance also. The true positives and true negatives each respectively decrease by 29 and 303 samples. The false positives increase by 29 samples. However, we see an improvement in the false negatives, with the biased models doing substantially better than the unbiased one by 282 samples. Thus, we get an improvement only in the false positives.

5 Discussion

With respect to results overall, we see that the biased models perform better than classical conceptor classification. On all 61 phoneme classes, the classification accuracies for all three biased models are better by around 2.5%. If the silence marker /h#/ is removed, the models show an improvement of around 2%. The absolute accuracies of the biased models are around 30.6% and 23.5%, and the unbiased model 28.2% and 21.9% respectively in these situations (Table 4.1).

Previous work on phoneme classification on the TIMIT data set have not provided a standardised method for testing classifiers. Different publications use different pre-processing and testing methodologies, thus it is difficult to directly compare this work to theirs (discussed further in sections 5.3.1-5.3.3). For example, Cao & Fan (2010) claim a mean phoneme classification accuracy of 93.1% – however, this is based on pre-segmented phoneme instances of just 5 predetermined phoneme classes.

The highest accuracy found in literature during this project for *windowed* classification was attained by Graves & Schmidhuber (2005), with a mean reported test accuracy of 73.2%. They used a BLSTM model to achieve this, with information flowing both forwards and backwards in time. They also report relatively high accuracies with normal LSTMs (70.2%), 'bidirectional' RNNs (65.3%) and normal RNNs (61.9%).

Note that they test on the *entire* TIMIT test set, whereas this work used only a subset. Additionally, they group phoneme labels to create a 43 class classification task instead of testing on all 61 phonemes. Thus this comparison only qualitative.

5.1 Within-Group Behaviour

Amongst phoneme groups, we see notable improvements in the biased models for some phoneme groups and decreases in others. For the Open Vowel and Nasal groups, there seems to be a effective increase in performance, and in the Closed Plosive and sh-Groups.

This could be explained linguistically: since we are working with MFCC data, which is in a transformed frequency domain, better performance may be expected in those phonemes which are better represented in the frequency domain. Vowels have very distinct formants (i.e. primary frequencies and overtones), whereas the fricatives and plosives tend to have noisy, non-distinct frequency values associated with them due to quite turbulent airflow (Huang et al., 2001). This might present in these conceptors as, respectively, information distributed amongst higher and lower singular values (though without further analysis, this is just conjecture). There may be more overlap between noisy plosive and fricative conceptors than those associated with vowels and nasals, and thus the biasing will not be as effective, or even be detrimental to the final model.

Intuitively with respect to the uncertainty, we may be able to distinguish vowels and nasals better than fricatives and plosives simply due to their frequencies respectively overlapping less and more. However, a more thorough analysis of this must be done in order to determine the changes in behaviour seen in these biased models.

We generally see similar results across all three biasing methods. However, we see a slight visual trend of the probability model doing worse than the other two, however, this has not been numerically quantified. The symbolic and fuzzy models seem to be on par for all phoneme groups. The only substantial difference seen in perhaps in the true negatives with respect to the Closed Plosive group of phonemes, where the symbolic and fuzzy methods have four times as many improved classifications compared to the probabilistic one.

However, these results may be task- and data set-specific. It is possible that top-down influence better assists in the disambiguation of some similar phoneme groups over others – in other words, purely symbolic, probabilistic or fuzzy prediction may be better for some groups than others. Additionally, these results do not consider the classification of silence phoneme /h#/, which may change the overall behaviour of the system.

5.2 Limitations

Only a fraction of the full TIMIT dataset was used in this project due to time and computation constraints. In a full study of bidirectional phoneme classification using the methods introduced here, more data such as from other dialects and female voices may be useful or be detrimental to the results of this experimental project. However, purely on an implementation level, the methods would need to be refined or paralleled to make such a system run quickly enough to be useful in real-world scenarios.

Additionally, hyper-parameter optimisation and cross-validation was not carried out for the same reason. This leaves much room for future research in this direction, for example, using these methods to increase overall classification performance.

5.3 Future Research

The current research aims to act as a stepping stone in using conceptors to bias reservoir networks to disambiguate similar signals. Thus, there may be several other applications of the core idea of this project that may be implemented elsewhere.

5.3.1 Reduced phoneme set

Previous attempts at classifying the TIMIT dataset often used a reduced phoneme set by clustering similar phonemes (such as plosives and their closed counterparts) and not considering phonemes that were not linguistically accepted (for example, three types of silence phonemes). For example, Graves & Schmidhuber (2005) use a reduced phoneme set of 43 phonemes, far fewer than the 61 used here. Cao & Fan (2010) use an ever smaller 5-class classification task on predetermined, phonetically unambiguous, phoneme classes. This may assist in better reported accuracies of classification. If the biasing methods introduced here are to be used competitively or in order to validate classification systems, similar data sets and pre-processing must be used.

5.3.2 Snippets versus Segmentation

The audio data in the TIMIT dataset is segmented and labelled with phoneme classes. The methods used here did not take make classifications with respect to this segmentation of phonemes in the audio data directly – rather, classification is on window-based audio-snippets. Audio-snippets may exist anywhere within the bounds of the segment. Training conceptors on audio-snippets generalises information about the phoneme, and longer temporal information such as changes in the start and end of the phoneme (which may help classification) is no longer explicitly present. Thus, a system that classifies on a full phoneme segment may have higher overall results than one that classifies audio-snippets (or other window-based structures).

The audio-snippets that lie on the boundary of labelled phonemes are also discarded, since these would arguably have two or more associated labels. Thus, a classification paradigm that uses these boundary conditions (which may contain important information about the transitions between phonemes) may also be fruitful.

5.3.3 Hyperparameter optimisation

Within the task of phoneme classification and time-series classification in general, there are several hyperparameters to consider when implementing the unbiased and biased classification methods described in this thesis. These include (but are not limited to):

- **Reservoir hyperparameters**: Spectral radius, choice of initial weights (i.e. the choice of sampling distribution), sparsity, initial network state.
- **Data preprocessing-centric hyperparameters**: Number of windows in an audio-snippet and amount of overlap between each of them, mel-frequency transform coefficients, filter banks, window lengths and strides.
- **Conceptor training hyperparameters**: Choice of initial aperture α, or alternatively, choice of α_{opt} calculation.

This creates a very large search-space of hyperparameters that would require significant computing power and time to exhaustively search. However this is still an important consideration in order to get optimal performance and/or a rigorous analysis of system behaviour.

5.3.4 Regulating Biasing

In order to understand the behaviour of the three biasing methods better, an analysis of the top-down biasing towards each phoneme class at each time step is required. This may be in the form, for example, of analysing how $\gamma(\Psi)$ relates to the system's behaviour over time. This may help elucidate how we may combine two sets of information with different semantics – *symbolic*, with human-comprehensible meaning pointing to abstract concepts or the relations between them, and *sub-symbolic*, arising from the relatively unclear non-linear dynamics of a neural network.

Notably, it is an open question as to how important the top-down model should be with respect to the bottom-up perception for a given task. Since biasing influences the prediction of the next audio-snippet based on the current, a 'bad' top-down prediction will bias the reservoir *away* from a potentially correct phoneme classification. This has the potential to cascade – a bad prediction now may influence decisions several steps into the future.

Overall, if the biasing towards a particular phoneme is too strong, the system may lose all benefits of reservoir computing, such as better noise-robustness, and fall back towards a purely rule-based predictor. Biasing the excitations too strongly towards a particular expected class will result in a high evidence value for that class, and thus, the top-down model will dictate behaviour of the system. The reverse is true as well – not biasing enough will lose the benefits of context that the top-down models provide, and move back towards classical conceptor classification. Thus, between these two flows of information, a balance must be achieved.

5.3.5 Uncertainty function Ψ and Aperture Relation $\gamma(\Psi)$

The function Ψ and the function used to compute the aperture adaptations $\gamma(\Psi)$ (discussed in section 1.9) are both core to the biasing models presented here. In essence, the function changes how important the higher singular values of the data (the ones with higher variance) compared to lower singular values (the ones with lesser variation). With this interpretation, the behaviour of the functions Ψ and $\gamma(\Psi)$ are not necessarily restricted – perhaps the differences between two similar phonemes /m/ and /n/ lie in the lower singular values, and other pairs may have their differences distributed elsewhere. The functions of Ψ chosen here, corresponding to conditional probabilities and fuzzy membership values are derived from anthropocentric intuitions of uncertainty-quantified symbolic reasoning. The optimal function may be completely different, either for the system or for actual human thought.

Thus, an interesting future study may be to find an optimal Ψ and analyse if its behaviour matches up with human intuition of uncertainty. Additionally, it may be interesting to study this it relates to apertures with a function $\gamma(\Psi)$, and see if aperture adaptation and uncertainty can indeed be analogous in conceptors.

5.3.6 More Context

Natural language often has long-term dependencies (Huang et al., 2001) that a first-order Markov matrix may not be able to capture. Thus, a better performing model may be created by enabling longer term context, for example, by assuming a higher-order Markov property. However, this will lead to an exponentially larger Markov matrix, and an approximation for the same may be necessary.

Additionally, analogously to the Ganong effect (Ganong, 1980), for an more offline classification task, or an online task where the system is allowed to alter previous predictions, future phonemes may also be used as context for previous ones in order to disambiguate similar sounds.

5.3.7 Other Conceptor Classification Paradigms

Jaeger (2014) describes several conceptor varieties, though two are used here for different purposes. The classification conceptors $\mathbf{C}^{\mathcal{P}}_{class}$ are used since they consist of high-dimensional temporal information of both the excitations and the signal, which is useful for comparing the shape of trajectories in the training set and those of new, test signals.

The biasing conceptors $\mathbf{C}^{\mathcal{P}}_{bias}$ are used to bias the reservoir since they represent a subspace of excitations. Other conceptors and architectures using them have been posited to be used in classification or other signal-related tasks, for example, the *autoconceptor*. Thus, it may be fruitful to explore the idea of reservoir biasing using other kinds of conceptor architectures.

5.3.8 Other Neural Networks and Tasks

Various natural language tasks other than phoneme transcription may benefit from top-down contextual information. Humans use context regularly in these kinds of tasks too. This includes tasks such as speed-to-text translation, speaker recognition, named entity recognition, and parts-of-speech tagging (Olsson, 2009).

Since reservoirs may be used for tasks that other RNN architectures are used for (Jaeger, 2014), the method outlined in this paper lends itself well to being extended into other time-series tasks that might leverage symbolic context. Non-linguistic temporal tasks such as stock market or weather prediction and control tasks may also presumably benefit from top-down contextual information flow as well.

Benefits of controlling or biasing neuronal excitations may prove to be useful in other RNNs too, including those trained with typical training routines (such as backpropagation through time), and not just in an untrained reservoir. This may be useful to fine-tune the predictions of an existing, trained network. Additionally, the idea of a conceptor is general one, and has been extended to non-reservoir networks such as feed-forward neural networks (He & Jaeger, 2018). Thus, it may be interesting to explore how context may be used in non-temporal tasks such as image recognition and segmentation.

6 Conclusion

The robustness of human context-enhanced perception, where symbolic top-down information interacts with low-level perceptual pathways, is not yet fully replicated in artificial neural networks. This project introduced three related methods of doing so, by biasing a reservoir network with conceptors that represent different symbols. The three methods are inspired by human symbolic, probabilistic and fuzzy reasoning methods, and a tentative mathematical relationship between them and conceptors were introduced. These method were used in an online learning task of automatic phoneme transcription. Though a more thorough analysis is required to determine the generality of its effectiveness, these methods show promise with fewer misclassifications amongst phonemes in similar-sounding phoneme groups compared to an unbiased model.

7 Acknowledgements

I thank Guillaume Pourcel and Steven Abreu for their supervision and continual guidance throughout the project. Additionally, I would like to thank Joris Peters, Daniel Woonings and especially David Coslar for our long conversations about conceptors that helped me gain insight and inspiration in this field. I wholeheartedly thank Ruhi Mahadeshwar, Andreea Minculescu, Pradnesh Mahadeshwar, Shania Sinha and my parents for coming on this journey with me.

References

Abadi, M. (2016). TensorFlow: Learning Functions at Scale. In *Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming.*

Abraham, A. (2005). Adaptation of Fuzzy Inference System Using Neural Learning. In N. Nedjah & L. d. Macedo Mourelle (Eds.), *Fuzzy Systems Engineering: Theory and Practice* (pp. 53–83). Springer. doi: 10.1007/11339366_3

Bellman, R., Kalaba, R., & Zadeh, L. (1966). Abstraction and Pattern Classification. *Journal of Mathematical Analysis and Applications*, *13*(1), pp. 1–7.

Bengio, Y., Lamblin, P., Popovici, D., & Larochelle, H. (2006). Greedy Layer-wise Training of Deep Networks. *Advances in Neural Information Processing Systems*, *19*.

Bermúdez, J. L. (2014). *Cognitive Science: An Introduction to the Science of the Mind.* Cambridge University Press.

Boden, M. A. (2014). Chapter 4: GOFAI. In K. Frankish & W. Ramsey (Eds.), *The Cambridge Handbook of Artificial Intelligence.* Cambridge University Press.

Cao, J., & Fan, G. (2010). Signal classification using random forest with kernels. In *2010 sixth advanced international conference on telecommunications* (pp. 191–195).

Ganchev, T., Fakotakis, N., & Kokkinakis, G. (2005). Comparative Evaluation of Various MFCC Implementations on the Speaker Verification Task. In *Proceedings of the SPECOM* (Vol. 1, pp. 191–194).

Ganong, W. F. (1980). Phonetic Categorization in Auditory Word Perception. *Journal of experimental psychology: Human perception and performance*, *6*(1), 110.

Garofolo, J. S., Lamel, L. F., Fisher, W. M., Fiscus, J. G., & Pallett, D. S. (1993). DARPA TIMIT Acoustic-phonetic Continous Speech Corpus CD-ROM. NIST speech disc 1-1.1. *NASA STI/Recon Technical Report*, *93*, 27403.

Gianakas, S. P., & Winn, M. (2016). Exploiting the Ganong Effect to Probe for Phonetic Uncertainty Resulting from Hearing Loss. *Journal of the Acoustical Society of America*, *140*, 3440.

Graves, A., & Schmidhuber, J. (2005). Framewise Phoneme Classification with Bidirectional LSTM Networks. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.* (Vol. 4, pp. 2047–2052).

Gwilliams, L., Linzen, T., Poeppel, D., & Marantz, A. (2018). In Spoken Word Recognition, the Future Predicts the Past. *Journal of Neuroscience*, *38*(35), 7585–7599.

He, X., & Jaeger, H. (2018). Overcoming Catastrophic Interference Using Conceptor-aided Backpropagation. In *International Conference on Learning Representations.*

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer Feedforward Networks are Universal Approximators. *Neural Networks*, *2*(5), 359–366.

Huang, X., Acero, A., Hon, H.-W., & Reddy, R. (2001). *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development.* Prentice Hall PTR.

Jaeger, H. (2014). Controlling Recurrent Neural Networks by Conceptors. *arXiv preprint arXiv:1403.3369*.

Kosko, B., & Burgess, J. C. (1998). *Neural Networks and Fuzzy Systems.* Acoustical Society of America.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems*, *25*.

Lyons, J., Wang, D., Gianluca, H. S., Mavrinac, E., Gaurkar, Y., Watcharawisetkul, W., ... others (2020). James Lyons/python speech features: Release v0. 6.1. *Zenodo*, *10*. doi: 10.5281/zenodo.3607820

Mittal, S., Lamb, A., Goyal, A., Voleti, V., Shanahan, M., Lajoie, G., ... Bengio, Y. (2020). Learning to Combine Top-down and Bottom-up Signals in Recurrent Neural Networks with Attention over Modules. In *International Conference on Machine Learning* (pp. 6972–6986).

Mossakowski, T., Glauer, M., & Diaconescu, R. (2018). Towards Logics for Neural Conceptors. *AITP 2018*.

Olsson, F. (2009). *A literature survey of active machine learning in the context of natural language processing* (1st ed.). Swedish Institute of Computer Science.

Rauss, K., & Pourtois, G. (2013). What is bottom-up and what is top-down in predictive coding? *Frontiers in psychology*, *4*, 276.

Yildiz, I. B., Jaeger, H., & Kiebel, S. J. (2012). Re-visiting the Echo State Property. *Neural Networks*, *35*, 1–9.

Zadeh, L. A. (1975). Fuzzy Logic and Approximate Reasoning. *Synthese*, *30*(3), 407–428.

Zadeh, L. A. (1996). Fuzzy Sets. In *Fuzzy sets, fuzzy logic, and fuzzy systems: Selected papers by Lotfi A Zadeh* (pp. 394–432). World Scientific.

A Appendices

A.1 List and Distribution of Phonemes

Table A.1 shows the list and distribution of phonemes found in the training and test sets defined in section 2. Figures A.1 and A.1 how this data graphically. We see a large non-uniformity in the distribution of the phonemes in both training and test sets, notably, the 'h#' phoneme label (used to indicate silence at the start and end of sentence) is greatly over-represented. However, it should be noted that both the training and testing distributions are visually very similar. For all of the details on this list, please refer to Garofolo et al. (1993).

A.2 Generating a Markov Matrix

In order to facilitate the symbolic and probabilistic biasing methods, we need to analyse the contexts of the occurrences of the phonemes themselves. For now, let us assume the Markov property within the data: the phoneme class at time $t+1$ is only dependant on the current one at time t. Note that this is not totally realistic, since often phonemes can have longer-term dependencies (for further discussion on this, see section 5.3.6).

With this assumption, the probabilistic biasing method described in section 3.4.2 relies on the calculation of the conditional probabilities of an audio-snippet of phoneme class p_j occurring after one with class p_i, i.e. how *probable* is p_j given p_i. Additionally, the symbolic method in section 3.4.1 can be interpreted as given two audio-snippets with phoneme classes p_i and p_j, we check in the training data if p_j ever follows from p_i – that is, check if the occurrence of p_j is *possible* after p_i with given previous experience. This is equivalent to checking if the condition probability is greater than zero.

Let the total number of occurrences of the sequence (p_i, p_j) (p_j occurring directly after p_i) be $N_{(p_i,p_j)}$, and the total number of occurrences of p_i be N_{p_i}. Thus the conditional probability for each pair of phonemes in the set of phoneme classes can be calculated as:

$$P(p_j|p_i) = \frac{P(p_j \cap p_i)}{P(p_i)} \tag{A.1}$$

$$= \frac{N_{(p_i,p_j)}}{N_{p_i}} \tag{A.2}$$

To reiterate, $\mathcal{P}$ is defined as the ordered set of phoneme classes seen in the training data. We may organise these conditional probabilities in a $|\mathcal{P}| \times |\mathcal{P}|$ matrix, whose numerical entry at position (i, j) represents the conditional probability $P(p_j|p_i)$. This is known as a Markov matrix. In this context, the phoneme classes may be termed as *states*, and their conditional probabilities as *state transitions* from one phoneme to another. Given the data set, $|\mathcal{P}| = 61$, thus, the Markov matrix is of shape 61×61.

A.3 Fuzzy Membership Prediction using a Multi-layer Perceptron

Let us assume that there exists a static function $\mathcal{F} : \mathbb{R}^{61} \to \mathbb{R}^{61}$ that takes the fuzzy membership values of all phonemes input as a vector at time t and outputs the same at time $t+1$. Thus, a multi-layer perceptron (MLP) was used to approximate this function, since MLPs are known to be universal function approximators (Hornik et al., 1989). For this, a small model was created using Tensorflow (Abadi, 2016). Since we assume the Markov property, no previous information before the current time step should influence the future: thus, an MLP was chosen, and not a temporal function approximating neural network such as a recurrent neural network. This MLP is notated as $\mathcal{F}$.

- **Data**: After training all classification conceptors $\mathbf{C}^{\mathcal{P}}_{class}$ as described in section 3.2, all samples in the training data were run through the reservoir once more, and the evidence values for each of the phoneme classes were computed using the procedure described in section 1.3 under *Testing*. These were then normalised to form fuzzy membership values as described in Eq. 1.14 by dividing the value by 325 (the dimension of each conceptor in $\mathbf{C}^{\mathcal{P}}_{class}$). Let the number of audio-snippets in the training data be $S = 79553$. Thus the membership values were collected sequentially in a matrix X_{all} of size $S \times |\mathcal{P}| = 79553 \times 61$.

  The inputs to the MLP were created as a matrix X_{train}, which consisted of all samples in X_{all} except the very last, which gives X_{train} the shape $(S-1) \times 61$. The corresponding targets to these inputs were the membership values that corresponded to exactly one time

Phoneme	**# Training Set**	**# Test Set**	**Phoneme**	**# Training Set**	**# Test Set**
aa	3817	659	ix	3428	683
ae	6239	780	iy	4978	944
ah	1769	387	jh	597	93
ao	3097	689	k	1859	266
aw	1110	88	kcl	2929	347
ax	1381	227	l	2940	573
ax-h	93	17	m	1993	360
axr	2156	386	n	2990	383
ay	3421	623	ng	616	106
b	118	20	nx	122	33
bcl	1028	224	ow	2766	498
ch	651	122	oy	1233	207
d	390	55	p	826	122
dcl	2143	265	pau	2008	341
dh	697	119	pcl	1541	167
dx	430	87	q	1989	318
eh	3175	670	r	3002	515
el	811	147	s	8111	1288
em	131	7	sh	2484	369
en	591	33	t	1482	279
eng	31	0	tcl	2824	411
epi	504	103	th	599	96
er	1838	410	uh	379	103
ey	2740	477	uw	844	123
f	2366	482	ux	1681	273
g	400	65	v	1046	183
gcl	852	142	w	1627	354
h#	19165	3195	y	1031	186
hh	509	86	z	2800	483
hv	660	116	zh	113	13
ih	3035	523			

Table A.1: List of phonemes and their total occurrences in terms of number of audio-snippets in the training and test sets.

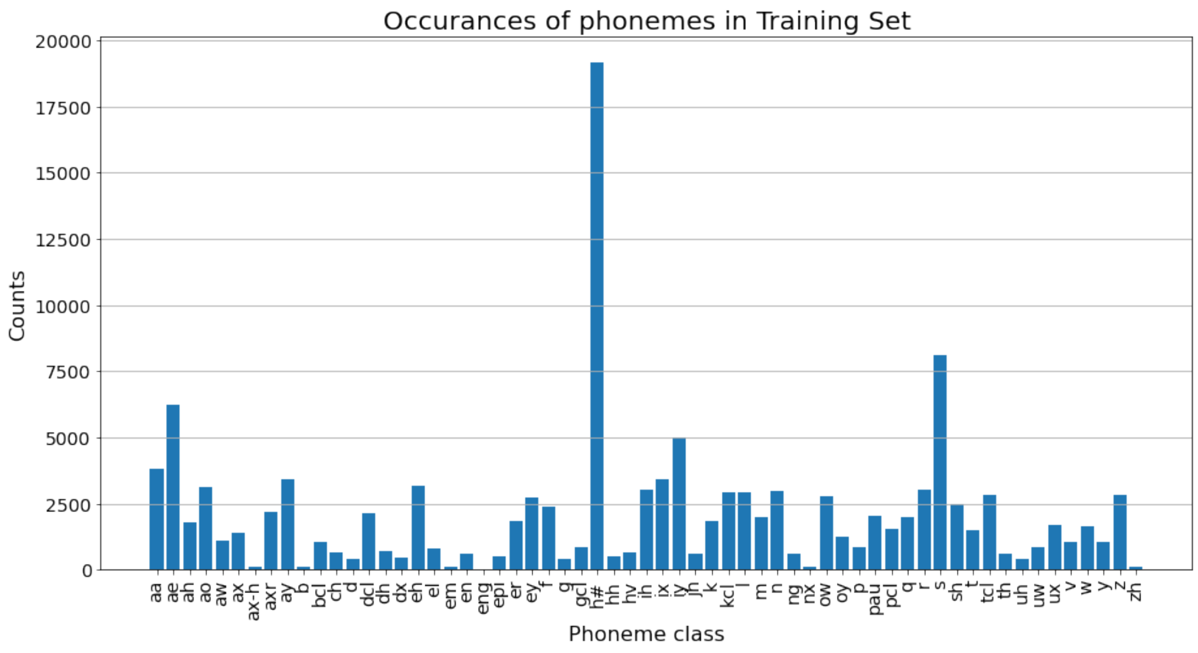


Figure A.1: Distribution of phonemes classes amongst audio-snippets in training set.

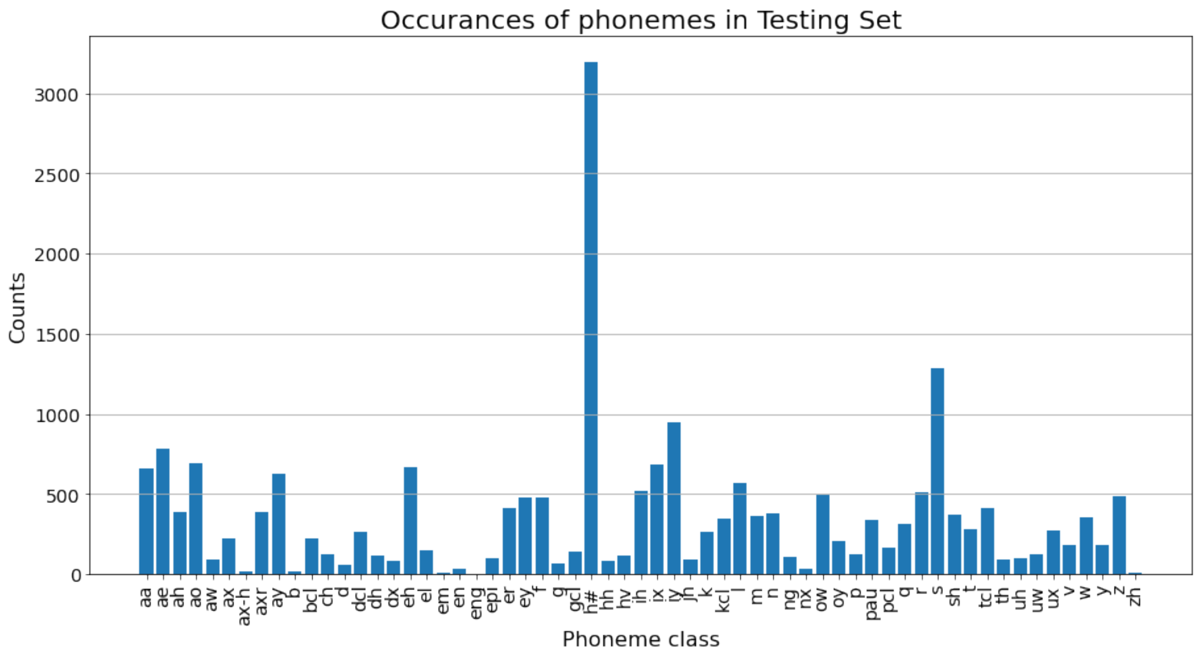


Figure A.2: Distribution of phonemes classes amongst audio-snippets in testing set.

step after the input. Thus, the training targets were collected in a matrix Y_{train} with shape $(S-1) \times 61$, which consisted of all samples in X_{all} apart from the first sample. Thus, for all $x_i \in X_{train}$, $y_i = x_{i+1}$, where $i \in \{1, ..., S-1\}$ is an index.

The pairs (x_i, y_i) were shuffled before training, and a random sample of 20% of the training set (15911 input-target pairs) was used as a validation set.

- **Architecture**: The MLP $\mathcal{F}$ consisted of two hidden layers of 128 neurons each, with an input size of 61 and an output size of 61. The two hidden layers used rectified linear units (ReLUs) as non-linear activation functions applied to them. The ReLU function is defined as:

$$\text{ReLU}(x) = \max(0, x) \tag{A.3}$$

Since this task can be seen as a regression task, the linear activation function is was first considered as the activation for the output layer. However, the outputs of this regression task is bounded – since we expect fuzzy membership values to be in the range $[0, 1]$, the sigmoid activation function σ was used as an alternative:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{A.4}$$

Due to an MLP's inherent non-linearity, out-of-distribution test examples may lead to the prediction of a particular membership value being greater than 1 or less than 0. Thus, the sigmoid activation function acts as a a 'squashing' function, having a range of $[0, 1]$ itself. Thus it was used as the activation function of the output layer of the MLP.

- **Training**: The model was trained using backpropagation, with the Adam optimiser, (with is a stochastic-gradient descent variant), with a learning rate of 0.0001. Note that the default values of Tensorflow 2 of the Adam optimiser was used during training: $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-7}$. For more details, refer to Abadi (2016). The loss function used was mean squared error, since we attempt to minimise the distance between the target vector and the one predicted by the MLP in all dimensions.

The model was trained for 15 epochs with a batch size of 128. The loss curves are shown in figure A.3. The trained model was saved to disk and loaded when necessary for the fuzzy conceptor classification paradigm.

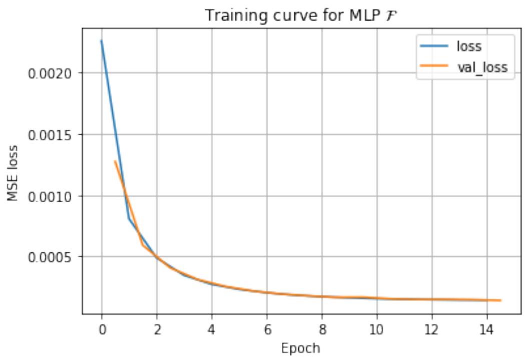


Figure A.3: Training curves for MLP $\mathcal{F}$. Training MSE loss is in blue, and validation loss is in orange. For details, refer to section A.3.

Thus, we achieve an approximation of a function $\mathcal{F}$ that outputs the set of fuzzy membership values of a set of 61 phoneme classes at time $t+1$ given the same at time t that was used in section 3.4.3.

Notes on Fuzzy-membership MLPs: There are a number of possible changes or improvements that may be explored in future research for approximating F:

- **Regularization**: Though figure A.3 visually suggests that the model is learning well and no overfitting is occurring, one might consider using a regularisation technique such as dropout to increase the generality of the network's predictions.

- **Hyper-parameter optimisation**: Hyper-parameters such as the learning rate, optimiser, number of epochs, etc. or even the MLP architecture itself were not optimised rigorously, but were tuned by hand until subjectively acceptable learning behaviour and performance was achieved. In order to get optimal results, a tuning method such as a grid search over the space of all hyper-parameters with k-fold cross validation would be ideal.

- **Removing Data Set Bias**: Finally, as we see in figures A.1 and A.2, the distributions of phonemes in the data is not uniform, and thus one would expect this to be reflected in the fuzzy membership value training data as well. This may lead to a bias in the predictions of the network. A more thorough investigation may be fruitful in understanding this bias and potentially taking it into account for the MLP's predictions.

A.4 Phoneme Conceptor Similarity

The heat map of similarities for all phoneme classes are shown in figure A.4. Phoneme labels on the $x-$ and $y-$axes are alphabetically ordered. The higher the similarity, the more red the region. Likewise, the lower the similarity, the more blue the similarity. We see that in general, all conceptors are relatively similar, with the most negative similarity around 0.625 between /eng/ and /z/. The similarity measure between the same conceptors is 1, which we see as a red diagonal on this heat map. We can also somewhat see grouping already: for example, the *Open Vowel* group in the top left of the graph.

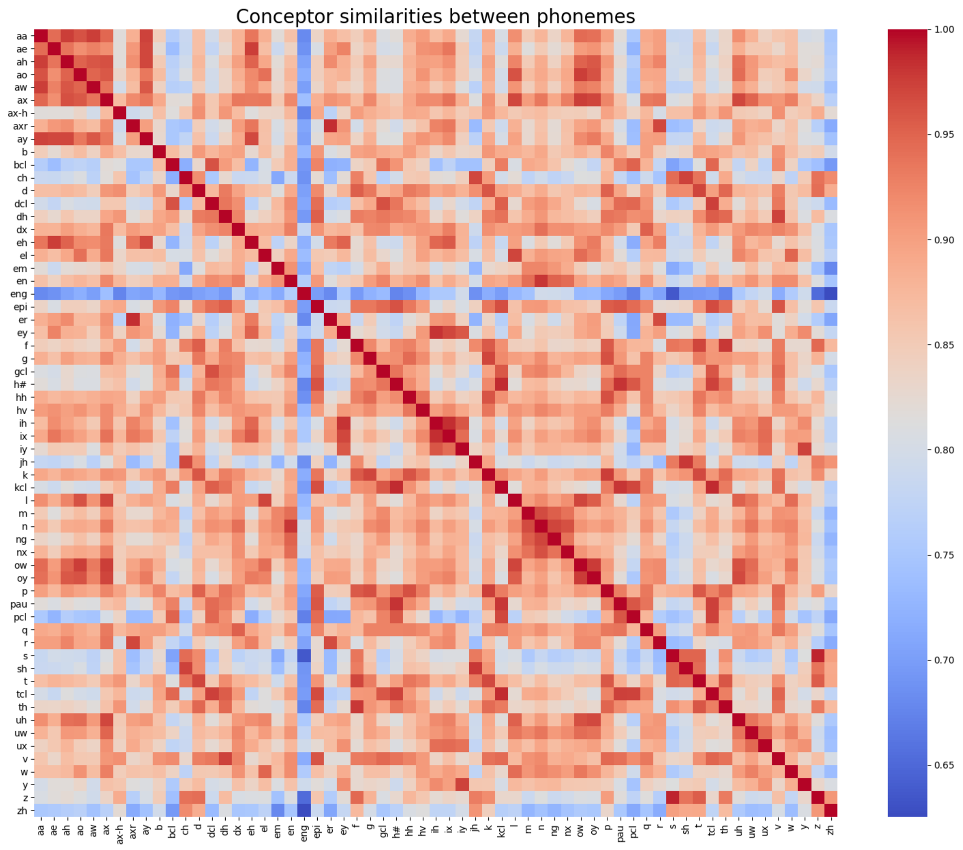


Figure A.4: Similarity heat map between phoneme conceptors $\mathbf{C}^{\mathcal{P}}_{bias}$. The conceptor similarity measure ranges from 0 to 1. Phoneme groups are tentatively seen here too – for example, members of the *Open Vowel* phoneme group can be seen in the top left corner of this heat map as a reddish square, since all their conceptors are similar to one another. The members of this group also subjectively sound very similar to one another.