A Bayesian Approach to Recurrence in Neural Networks

Philip N. Garner, Sibo Tong

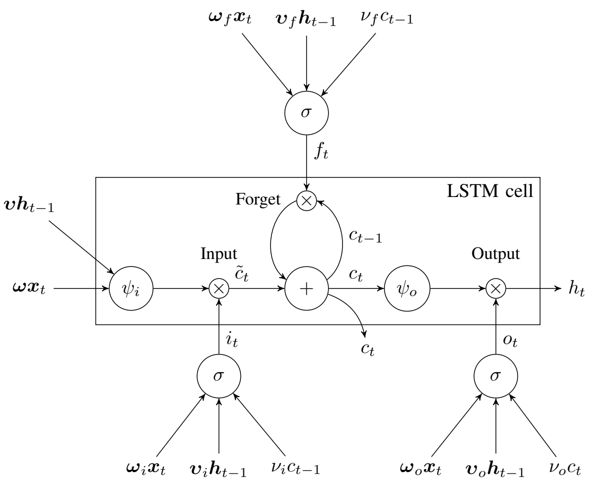
***Abstract*—We begin by reiterating that common neural network activation functions have simple Bayesian origins. In this spirit, we go on to show that Bayes's theorem also implies a simple recurrence relation; this leads to a Bayesian recurrent unit with a prescribed feedback formulation. We show that introduction of a context indicator leads to a variable feedback that is similar to the forget mechanism in conventional recurrent units. A similar approach leads to a probabilistic input gate. The Bayesian formulation leads naturally to the two pass algorithm of the Kalman smoother or forward-backward algorithm, meaning that inference naturally depends upon future inputs as well as past ones. Experiments on speech recognition confirm that the resulting architecture can perform as well as a bidirectional recurrent network with the same number of parameters as a unidirectional one. Further, when configured explicitly bidirectionally, the architecture can exceed the performance of a conventional bidirectional recurrence.**

I. Introduction

In signal processing and statistical pattern recognition, recurrent models have been ubiquitous for some time. They are perhaps exemplified by two cases: the state space filter of Kalman [1], [2] is appropriate for continuous states; the hidden Markov model (HMM) [3], [4] for discrete states. Both of these approaches can be characterised as being statistically rigorous; each has a forward-backward training procedure that arises from a statistical estimation formulation.

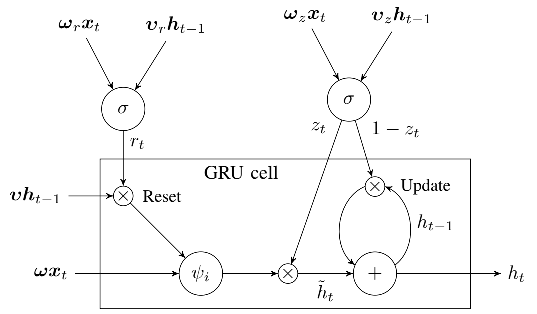
Recurrence is also important in modern deep learning. The foundations were laid shortly after the introduction of the multi-layer perceptron (MLP) [5], [6] with the back-propagation through time algorithm [5], [7]. Such architectures can be difficult to train; some of the difficulties were addressed by the long short-term memory (LSTM) of Hochreiter and Schmidhuber [8]. The LSTM was subsequently modified by Gers et al. [9] to include a forget gate, and by Gers et al. [10] to include peephole connections. The full LSTM is illustrated in figure 1.

The LSTM's concept of gates has since been used in the gated recurrent unit (GRU) of Cho et al. [11], and remains important. In GRU, the input and forget gates are combined into a single operation, and the output gate is applied to the recurrent part of the input instead. It is illustrated in figure 2. The GRU has also been modified: In a minimally gated unit (MGU), Zhou et al. [12] replace the reset gate with a



$\boldsymbol{\omega}_f \boldsymbol{x}_t$
$\boldsymbol{v}_f \boldsymbol{h}_{t-1}$
$\nu_f c_{t-1}$
σ
f_t
LSTM cell
Forget
$\boldsymbol{v}\boldsymbol{h}_{t-1}$
c_{t-1}
Input
Output
$\tilde{c}_t$
c_t
$\boldsymbol{\omega}\boldsymbol{x}_t$
ψ_i
ψ_o
h_t
i_t
c_t
o_t
σ
σ
$\boldsymbol{\omega}_i \boldsymbol{x}_t$
$\boldsymbol{v}_i \boldsymbol{h}_{t-1}$
$\nu_i c_{t-1}$
$\boldsymbol{\omega}_o \boldsymbol{x}_t$
$\boldsymbol{v}_o \boldsymbol{h}_{t-1}$
$\nu_o c_t$

Fig. 1. The long short term memory of [8]. Non-linearities ψ are taken to be tanh.



$\boldsymbol{\omega}_r \boldsymbol{x}_t$
$\boldsymbol{v}_r \boldsymbol{h}_{t-1}$
$\boldsymbol{\omega}_z \boldsymbol{x}_t$
$\boldsymbol{v}_z \boldsymbol{h}_{t-1}$
σ
σ
r_t
z_t
$1 - z_t$
GRU cell
$\boldsymbol{v}\boldsymbol{h}_{t-1}$
Reset
Update
h_{t-1}
$\boldsymbol{\omega}\boldsymbol{x}_t$
ψ_i
$\tilde{h}_t$
h_t

Fig. 2. The gated recurrent unit of [11]. As in the LSTM, the non-linearity ψ is usually tanh.

signal from the update gate; in the notation here, r_t is replaced by $1 - z_t$. Ravanelli et al. [13], [14] remove the reset gate altogether in their light GRU (Li-GRU), equivalent to setting $r_t = 1$.

Notice that the LSTM and GRU implicitly define three types of recurrence:

1) A *unit-wise* recurrence, exemplified by the constant error carousel (CEC, forget loop) of the LSTM or the GRU update loop.
2) A *layer-wise* recurrence, being the vector loop $\boldsymbol{h}_{t-1}$ from output to input.
3) A *gate* recurrence, being the vector loop from output to gate.

Several authors have noted the similarities between HMMs and (recurrent) networks. Bourlard and Wellekens [15] show

doi:10.1109/TPAMI.2020.2976978

that the two architectures can be made to compute similar probabilistic values. Bridle [16] shows that a suitably designed network can mimic the *alpha* part of the forward-backward algorithm. Bridle also points out similarities between the back-propagation (of derivatives) in the training of MLPs and the backward pass in HMMs. With the bidirectional recurrent neural network (BiRNN), in contrast to *seeking* relationships, Schuster & Paliwal [17] *imposed* the backward relationship between HMMs and MLPs by means of a second recurrence relationship running in the opposite direction. This was in fact to explicitly allow the network to take account of "future" observations. The natural substitution of LSTMs for the same purpose was described by Graves & Schmidhuber [18] resulting in the bidirectional LSTM (BLSTM or BiLSTM); this type of network remains the state of the art in several fields.

Putting aside the concept of recurrence, probabilistic interpretations of feed-forward MLPs are well known. Although the sigmoid is usually described as being a smooth (hence differentiable) approximation of a step function, its probabilistic origin was pointed out by Bridle [19], and is well known to physicists via the Boltzmann distribution. It has also been shown that the training process yields parameters that make sense in a statistical sense; this is evident from the work of Richard & Lippman [20], summarising work such as that of [15], and most thoroughly by MacKay [21]–[23] in papers that constituted his PhD thesis, later popularised by Bishop [24].

In the present paper, we build on this latter body of work, recalling that several MLP concepts have sound Bayesian origins. We show that this implies a natural probabilistic recurrence, leading to an architecture similar to the GRU [11]. We go on to show that, because the derivation is probabilistic, a backward recursion is also evident; this without the explicit extra backward recurrence of the BiRNN architectures described above. Experiments on standard speech recognition tasks show that this recurrent architecture can yield performance near indistinguishable from that of BiRNNs. Finally, we show that when this implicit bidirectional network is doubled up to be explicitly bidirectional, it can exceed the performance of BiRNNs.

II. Background

A. Bayesian interpretation of MLP units

We begin by making explicit a relationship, pointed out by Bridle [19], between Bayes's theorem and the sigmoid activation; we show that the same relationship also applies to ReLU (rectifying linear unit) activations.

Say we have an observation vector, $\boldsymbol{x}$, and we want the probability that it belongs to class i, where $i \in \{1, 2, \ldots, C\}$. The Bayesian solution is

$$P(c_i \mid \boldsymbol{x}) = \frac{p(\boldsymbol{x} \mid c_i)\, P(c_i)}{\sum_{j=1}^{C} p(\boldsymbol{x} \mid c_j)\, P(c_j)}, \tag{1}$$

where c_i refers to the event that the class takes value i, and $\boldsymbol{x}$ refers to the event that the observation random variable takes value $\boldsymbol{x}$.

If we take the observations to be from multivariate Gaussian distributions then, in the two class case, $C = 2$,

$$P(c_1 \mid \boldsymbol{x}) = \frac{1}{1 + \exp\left(-(\boldsymbol{\omega}^{\mathsf{T}}\boldsymbol{x} + \upsilon)\right)}, \tag{2}$$

where

$$\boldsymbol{\omega}^{\mathsf{T}} = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^{\mathsf{T}}\boldsymbol{\Sigma}^{-1} \tag{3}$$

$$\upsilon = \log P(c_1) - \log P(c_2) - \frac{1}{2}\left(\boldsymbol{\mu}_1^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_2\right), \tag{4}$$

and $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}$ are respectively the mean and covariance of the constituent Gaussians. The class priors in this case, $P(c_i)$, are taken to be constant and subsumed in the bias term. This is the commonly used sigmoid activation.

In the multi-class case, $C \geq 2$,

$$P(c_i \mid \boldsymbol{x}) = \frac{\exp\left(\boldsymbol{\omega}_i^{\mathsf{T}}\boldsymbol{x} + \upsilon_i\right)}{\sum_{j=1}^{C} \exp\left(\boldsymbol{\omega}_j^{\mathsf{T}}\boldsymbol{x} + \upsilon_j\right)}, \tag{5}$$

where

$$\boldsymbol{\omega}_i^{\mathsf{T}} = \boldsymbol{\mu}_i^{\mathsf{T}}\boldsymbol{\Sigma}^{-1} \tag{6}$$

$$\upsilon_i = \log P(c_i) - \frac{1}{2}\boldsymbol{\mu}_i^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i. \tag{7}$$

This is the softmax activation function introduced in [19].

A Gaussian assumption is appropriate for MLP inputs. However, hidden layers take inputs from previous layers with sigmoid outputs; their values are closer to beta distributions. If, instead of a Gaussian, the observations are assumed to follow independent beta distributions,

$$p(x) = \frac{1}{B(\alpha, \beta)} x^{\alpha-1}(1-x)^{\beta-1} \tag{8}$$

$$= \frac{1}{B(\alpha, \beta)} e^{(\alpha-1)\log(x)} e^{(\beta-1)\log(1-x)}, \tag{9}$$

where the second line emphasises that the beta is exponential family. With $\beta = 1$, we then have:

$$P(c_1 \mid \boldsymbol{x}) = \frac{1}{1 + \exp\left(-(\boldsymbol{\omega}^{\mathsf{T}}\log(\boldsymbol{x}) + \upsilon)\right)}, \tag{10}$$

with

$$\boldsymbol{\alpha}_j = (\alpha_{j,1}, \ldots, \alpha_{j,P})^{\mathsf{T}}, \tag{11}$$

$$\boldsymbol{\omega} = \boldsymbol{\alpha}_1 - \boldsymbol{\alpha}_2 \tag{12}$$

$$\upsilon = \log P(c_1) - \log P(c_2) \tag{13}$$

$$- \sum_{i=1}^{P}\left[\log B(\alpha_{1,i}, 1) - \log B(\alpha_{2,i}, 1)\right] \tag{14}$$

and P is the input dimension.

So, when a sigmoid output is used as the input to a subsequent layer, the value that makes sense under a beta assumption is its logarithm. Taking a logarithm of a sigmoid results in the softplus described by Dugas et al. [25] albeit for a different reason. Glorot et al. [26] show that the ReLU is a linear approximation to the softplus.

III. General probabilistic recurrence

In the previous section, we showed that the main activations used in MLPs have probabilistic explanations. In this spirit, we derive a recursive activation from a probabilistic point of view. At the outset, we expect the formulation to dictate the form of the recursion, removing otherwise ad-hoc aspects of standard techniques.

A. Conditional independence of observations

Let us assume that we have a (temporal) sequence of observations $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T$. Equation 1 becomes (abbreviated for the moment)

$$P\left(c_i \mid \boldsymbol{x}_T, \boldsymbol{x}_{T-1}, \ldots, \boldsymbol{x}_1\right) \propto p\left(\boldsymbol{x}_T \mid c_i, \boldsymbol{x}_{T-1}, \ldots, \boldsymbol{x}_1\right) P\left(c_i \mid \boldsymbol{x}_{T-1}, \ldots, \boldsymbol{x}_1\right). \quad (15)$$

If we then assume that all the $\boldsymbol{x}_t$ are conditionally independent given c_i, we have

$$P\left(c_i \mid \boldsymbol{x}_T, \boldsymbol{x}_{T-1}, \ldots, \boldsymbol{x}_1\right) \propto p\left(\boldsymbol{x}_T \mid c_i\right) P\left(c_i \mid \boldsymbol{x}_{T-1}, \ldots, \boldsymbol{x}_1\right). \quad (16)$$

This is a standard recursion where the posterior at time $t-1$ forms the prior for time t.

B. Application to MLP

More generally, say we have a matrix, $\boldsymbol{X}_T$, the rows of which are observation vectors $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T$. There is a corresponding matrix, $\boldsymbol{H}_T$, the rows of which are vectors $\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_T$. We assume each element $h_{t,i}$ of $\boldsymbol{H}$ represents a probability $P\left(\phi_i \mid \boldsymbol{X}_t\right)$ of the event that feature i exists in the observation sequence up to time t. Conversely, $1-h_{t,i} = P\left(\bar{\phi}_i \mid \boldsymbol{X}_t\right)$. Notice that, at this stage, ϕ_i is not time dependent; the feature exists (or not) for the whole sequence, with each observation in the sequence updating $P\left(\phi_i \mid \boldsymbol{X}_t\right)$. Now say that the probabilities $P\left(\phi_i \mid \boldsymbol{X}\right)$ are independent given some parameters, $\boldsymbol{\theta}$. So the joint probability is the product

$$P\left(\phi_1, \phi_2, \ldots, \phi_F \mid \boldsymbol{\theta}, \boldsymbol{X}_t\right) = P\left(\phi_1 \mid \boldsymbol{\theta}, \boldsymbol{X}_t\right) P\left(\phi_2 \mid \boldsymbol{\theta}, \boldsymbol{X}_t\right) \ldots P\left(\phi_F \mid \boldsymbol{\theta}, \boldsymbol{X}_t\right). \quad (17)$$

For a given feature, ϕ_i,

$$h_{t,i} = P\left(\phi_i \mid \boldsymbol{\theta}, \boldsymbol{X}_t\right) \quad (18)$$

$$= \frac{p\left(\boldsymbol{x}_t \mid \phi_i, \boldsymbol{\theta}, \boldsymbol{X}_{t-1}\right) P\left(\phi_i \mid \boldsymbol{\theta}, \boldsymbol{X}_{t-1}\right)}{\sum_{\phi_i} p\left(\boldsymbol{x}_t \mid \phi_i, \boldsymbol{\theta}, \boldsymbol{X}_{t-1}\right) P\left(\phi_i \mid \boldsymbol{\theta}, \boldsymbol{X}_{t-1}\right)} \quad (19)$$

$$= \frac{1}{1 + \frac{p\left(\boldsymbol{x}_t \mid \bar{\phi}_i\right)}{p\left(\boldsymbol{x}_t \mid \phi_i\right)} \cdot \frac{P\left(\bar{\phi}_i \mid \boldsymbol{X}_{t-1}\right)}{P\left(\phi_i \mid \boldsymbol{X}_{t-1}\right)}}, \quad (20)$$

where, in the final line and hereafter, we drop the conditioning on $\boldsymbol{\theta}$ for clarity. The final expression contains two fractional terms. The first of these follows from the conditional independence assumption above, and leads to the sigmoid of equations 2 and 10, but without the priors in the bias terms. Instead of being static, the priors form the second fractional term which is a multiplicative feedback

$$\frac{P\left(\bar{\phi}_i \mid \boldsymbol{X}_{t-1}\right)}{P\left(\phi_i \mid \boldsymbol{X}_{t-1}\right)} = \frac{1-h_{t-1,i}}{h_{t-1,i}} = \frac{1}{\operatorname{odds}(h_{t-1,i})} \quad (21)$$

If this were indeed included as an additive component of the bias in equations 2 or 10 then the fed back term would be

$$\log\left(\frac{h_{t-1,i}}{1-h_{t-1,i}}\right) = \operatorname{logit}(h_{t-1,i}) \quad (22)$$

$$= \log(h_{t-1,i}) - \log(1-h_{t-1,i}). \quad (23)$$

The logit and odds functions are illustrated in figure 3.

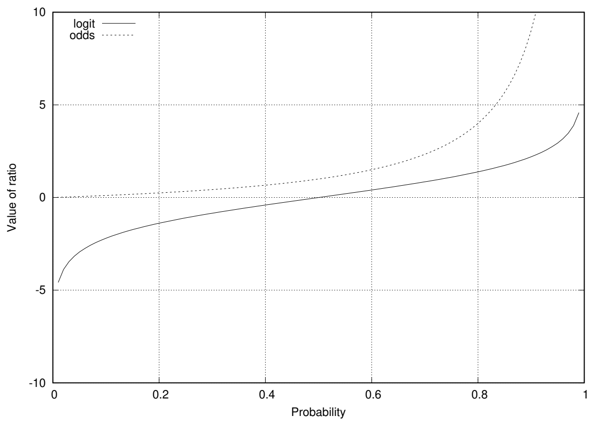


Fig. 3. Logit and odds curves.

IV. Probabilistic forget

The BRU described above carries the assumption that a feature is present (or not) in the entire input sequence. By contrast, we know from the LSTM that it is necessary to allow an activation to respond differently to different inputs depending on the context. In an LSTM this is achieved using gates. We show here that gates can be derived probabilistically.

Say that $P\left(\phi_i\right)$ is somehow dependent upon another variable indicative of context. For instance, if ϕ_i is indicative of a characteristic of a sentence, it is dependent upon the previous words in the sentence, but resets after a (grammatical) period, when the sentence changes. Say there is a binary state variable, ζ, where $\zeta = 1$ indicates the context remaining relevant, and $\zeta = 0$ indicates that it is not relevant. We can assign a probability, $z_t = P\left(\zeta_t = 1 \mid \boldsymbol{X}_t\right)$ and the inverse $(1-z_t) = P\left(\zeta_t = 0 \mid \boldsymbol{X}_t\right)$, where z_t is predicted by the network. It is then the prior (in equation 21) that depends on the context. $\boldsymbol{\phi}$ is now dependent upon the time index, t.

Note that the state variable can be defined for one or multiple features. In the following derivation, we assume only one feature, removing the need for an index. However, it is common for recurrence to use one variable per feature.

A. Unit-wise recursion

We first consider the case where the ϕ_i are taken to be independent; it is derived in equations 24–27 below, where p_i is the unconditional prior probability of feature i being present.

$$P\left(\phi_{t,i} \mid \boldsymbol{X}_{t-1}\right) = \sum_{\phi_{t-1,i}} \sum_{\zeta_{t-1}} P\left(\phi_{t,i} \mid \phi_{t-1,i}, \zeta_{t-1}, \boldsymbol{X}_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right) P\left(\zeta_{t-1} \mid \boldsymbol{X}_{t-1}\right) \tag{24}$$

$$\begin{aligned} &= P\left(\phi_{t,i} \mid \phi_{t-1,i}, \zeta_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right) P\left(\zeta_{t-1} \mid \boldsymbol{X}_{t-1}\right) \\ &\quad + P\left(\phi_{t,i} \mid \bar{\phi}_{t-1,i}, \zeta_{t-1}\right) P\left(\bar{\phi}_{t-1,i} \mid \boldsymbol{X}_{t-1}\right) P\left(\zeta_{t-1} \mid \boldsymbol{X}_{t-1}\right) \\ &\quad + P\left(\phi_{t,i} \mid \phi_{t-1,i}, \bar{\zeta}_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right) P\left(\bar{\zeta}_{t-1} \mid \boldsymbol{X}_{t-1}\right) \\ &\quad + P\left(\phi_{t,i} \mid \bar{\phi}_{t-1,i}, \bar{\zeta}_{t-1}\right) P\left(\bar{\phi}_{t-1,i} \mid \boldsymbol{X}_{t-1}\right) P\left(\bar{\zeta}_{t-1} \mid \boldsymbol{X}_{t-1}\right) \end{aligned} \tag{25}$$

$$\begin{aligned} &= 1 \times h_{t-1,i} z_{t-1} \\ &\quad + 0 \times (1 - h_{t-1,i}) z_{t-1} \\ &\quad + p_i h_{t-1,i} (1 - z_{t-1}) \\ &\quad + p_i (1 - h_{t-1,i})(1 - z_{t-1}) \end{aligned} \tag{26}$$

$$= (1 - z_{t-1}) p_i + z_{t-1} h_{t-1,i}, \tag{27}$$

Notice that the simplifications arise from the interaction of $\phi_{t,i}$, $\phi_{t-1,i}$ and ζ_{t-1}: context remaining relevant implies the feature should remain. So, for instance, the feature changing from not present to present when context is relevant has zero probability.

In a Kalman filter sense, $P\left(\phi_{t,i} \mid \boldsymbol{X}_{t-1}\right)$ is the predictor. The result is an intuitive linear combination of the previous output with a prior. In this paper, although we deal with a discrete state variable, we use the Kalman filter analogy because it is easier to follow. Nevertheless, a correspondence with *alpha*, *beta* and *gamma* probabilities will be evident to readers familiar with Markov models.

There is a question of initialisation. The first output corresponding to $t = 1$ should use the value $h_{0,i} = p_i$; thereafter, the value from the feedback loop can be taken.

At time $t = 1$, $\quad z_{t-1} = 0, \quad h_{t-1,i} = p_i$

At time $t = 2$, $\quad z_{t-1} = f_z(\boldsymbol{X}_{t-1}), \quad h_{t-1,i} = f_h(\boldsymbol{X}_{t-1})$

where $f_.(\cdot)$ is taken to mean "some function of". If $h_{t-2,i}$ is required, the same value as $h_{t-1,i}$ can be used. In turn, the fed back value (equation 21) is actually

$$\frac{1 - P\left(\phi_{t,i} \mid \boldsymbol{X}_{t-1}\right)}{P\left(\phi_{t,i} \mid \boldsymbol{X}_{t-1}\right)} = \frac{1}{\operatorname{odds}\left([1 - z_{t-1}]p_i + z_{t-1}h_{t-1,i}\right)}, \tag{28}$$

with the logarithm of the reciprocal being the additive term inside the exponential. This is illustrated in figure 4 where,

$$f(\cdot) = \operatorname{logit}\left([1 - z_{t-1}]p_i + z_{t-1}h_{t-1,i}\right). \tag{29}$$

In figure 4, note that the unit-wise recurrence is probabilistic, but an *ad-hoc* layer-wise and gate recurrence are also retained for comparison with a GRU. The $\boldsymbol{h}_{t-2}$ term in this and later cases arises to maintain a consistent definition of z_t across the LSTM, GRU and equation 72; we note that, in practice, the extra delay makes no difference in performance.

B. Discussion

The unit-wise recursion above was an attempt to formalise the "constant error carousel" (CEC) — the central recurrence — of the LSTM. Whilst the result is self consistent, in practice we find two difficulties:

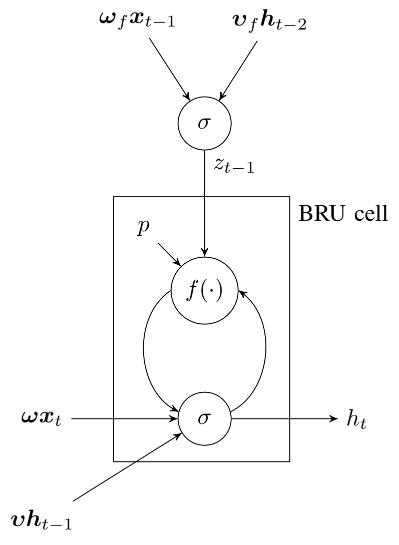

$\boldsymbol{\omega}_f \boldsymbol{x}_{t-1}$
$\boldsymbol{v}_f \boldsymbol{h}_{t-2}$
σ
z_{t-1}
BRU cell
p
$f(\cdot)$
$\boldsymbol{\omega} \boldsymbol{x}_t$
σ
h_t
$\boldsymbol{v} \boldsymbol{h}_{t-1}$

Fig. 4. A Bayesian recurrent unit incorporating a probabilistic forget gate.

1) The logit function of equation 29 causes instability in the training process. This is because it can tend to $\pm\infty$.
2) The formulation does not explain the layer-wise recursion around the whole layer of units.

In the following, we address both of these difficulties using approximations. We find that the resulting layer-wise recursion is both stable and more complete.

C. Layer-wise recursion

In contrast to the unit-wise recursion above, here we take the elements of $\boldsymbol{\phi}$ to be dependent, meaning the summation is over the whole vector. The main derivation is equations 30–32 below, The calculation can be rendered tractable if we model $P\left(\phi_{t,i} \mid \boldsymbol{\phi}_{t-1}, \zeta_{t-1}\right)$ as $\boldsymbol{\omega}_i^{\mathsf{T}} \boldsymbol{\phi}_{t-1}$, where $\boldsymbol{\omega}_i$ is a trainable vector and each element $\omega_{j,i}$ models the weight that $\phi_{t-1,j}$ has on $\phi_{t,i}$. The occurrence of $\phi_{t,i}$ is considered to be the weighted average of the occurrences of $\boldsymbol{\phi}_{t-1}$. This is an extension of unit-wise recursion where the occurrence of

$$P\left(\phi_{t,i} \mid \boldsymbol{X}_{t-1}\right) = \sum_{\boldsymbol{\phi}_{t-1}} \sum_{\zeta_{t-1}} P\left(\phi_{t,i} \mid \boldsymbol{\phi}_{t-1}, \zeta_{t-1}, \boldsymbol{X}_{t-1}\right) P\left(\boldsymbol{\phi}_{t-1} \mid \boldsymbol{X}_{t-1}\right) P\left(\zeta_{t-1} \mid \boldsymbol{X}_{t-1}\right) \tag{30}$$

$$\begin{aligned} &= P\left(\zeta_{t-1} \mid \boldsymbol{X}_{t-1}\right) \sum_{\boldsymbol{\phi}_{t-1}} P\left(\phi_{t,i} \mid \boldsymbol{\phi}_{t-1}, \zeta_{t-1}\right) P\left(\boldsymbol{\phi}_{t-1} \mid \boldsymbol{X}_{t-1}\right) \\ &\quad + P\left(\bar{\zeta}_{t-1} \mid \boldsymbol{X}_{t-1}\right) \sum_{\boldsymbol{\phi}_{t-1}} P\left(\phi_{t,i} \mid \boldsymbol{\phi}_{t-1}, \bar{\zeta}_{t-1}\right) P\left(\boldsymbol{\phi}_{t-1} \mid \boldsymbol{X}_{t-1}\right). \end{aligned} \tag{31}$$

$$\begin{aligned} &= z_{t-1} \sum_{\boldsymbol{\phi}_{t-1}} P\left(\phi_{t,i} \mid \boldsymbol{\phi}_{t-1}, \zeta_{t-1}\right) \prod_i P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right) \\ &\quad + (1 - z_{t-1}) \sum_{\boldsymbol{\phi}_{t-1}} P\left(\phi_{t,i} \mid \boldsymbol{\phi}_{t-1}, \bar{\zeta}_{t-1}\right) \prod_i P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right). \end{aligned} \tag{32}$$

$\phi_{t,i}$ only depends on $\phi_{t-1,i}$ and $\boldsymbol{\omega}_i$ is a one-hot vector with $\omega_{i,i} = 1$. Therefore, we have

$$\sum_{\boldsymbol{\phi}_{t-1}} P\left(\phi_{t,i} \mid \boldsymbol{\phi}_{t-1}, \zeta_{t-1}\right) \prod_i P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right) = \boldsymbol{\omega}_i^{\mathsf{T}} \boldsymbol{h}_t + c \tag{33}$$

where, $\boldsymbol{\omega}_i$ denotes the i^{th} column of $\boldsymbol{\omega}$ and $c = \sum_{j \in \{j \mid \omega_{j,i} < 0\}} \omega_{j,i}$. To understand the above equation, consider N independent lotteries, where N is the total number of nodes in a layer. The winning rate of the i^{th} lottery is h_i, the corresponding prize is ω_i. Now we buy each of the lottery once. The left side of the above equation actually calculate the expectation of the total prizes we can win from the lotteries by listing all the possibilities. On the other hand, each lottery is independent. Therefore, the expectation prize for i^{th} lottery is $\omega_i h_i$. The expectation of the total prizes we can get is then $\boldsymbol{\omega}_i^{\mathsf{T}} \boldsymbol{h}$.

In this sense, the recursion is parameterised by matrix $\boldsymbol{\omega}$. Given the fact that $\boldsymbol{\omega}_i^{\mathsf{T}} \boldsymbol{\phi}_{t-1}$ represents probabilities and the expectation of probabilities should be positive, it is sensible to constrain the L_1 norm of each column in $\boldsymbol{\omega}$ to 1 and add the bias term c. Thus, equation 32 can be written as

$$P\left(\phi_{t,i} \mid \boldsymbol{X}_{t-1}\right) = z_{t-1}(\boldsymbol{\omega}_i^{\mathsf{T}} \boldsymbol{h}_{t-1} + c - p_i) + p_i$$

This is illustrated in 5, where,

$$f(\cdot) = \operatorname{logit}\left(z_{t-1}(\boldsymbol{\omega}_i^{\mathsf{T}} \boldsymbol{h}_{t-1} + c - p_i) + p_i\right). \tag{34}$$

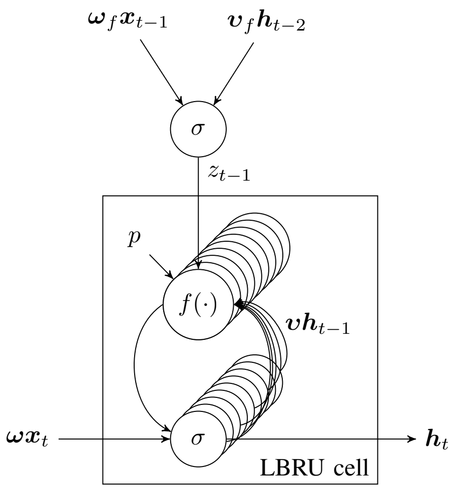
Note that in figure 5, the unit-wise and layer-wise recurrence are combined into a single probabilisitic recurrence. However, the ad-hoc gate recurrence is retained. With reference to figure 3, the function $\log\left(\frac{h}{1-h}\right)$ appears linear except for narrow regions close to 0 and 1. Since we are not aware of the distribution of h, we further approximate $\log\left(\frac{h}{1-h}\right) \approx \alpha h + \beta$, yielding

$$\log\left(\frac{P\left(\phi_{t,i} \mid \boldsymbol{X}_{t-1}\right)}{1 - P\left(\phi_{t,i} \mid \boldsymbol{X}_{t-1}\right)}\right) \approx z_{t-1}(\boldsymbol{\omega}_i^{\mathsf{T}} \boldsymbol{h}_{t-1} + c - p_i) + p_i + \beta, \tag{35}$$

where α is absorbed by $\boldsymbol{\omega}_i^{\mathsf{T}}$ and p_i. The range of α is $[4, +\infty)$. Therefore, we do not normalise $\boldsymbol{\omega}_i$ in the forward pass.

Substituting back into equation 20, that equation can be rewritten as:

$$\boldsymbol{h}_t = \sigma(\boldsymbol{\omega}_{ih} \boldsymbol{x}_t + \mathbf{b}_{ih} + \mathbf{z}_{t-1} \odot (\boldsymbol{\omega}_{hh} \boldsymbol{h}_{t-1} + \mathbf{b}_{hh})) \tag{36}$$



$\boldsymbol{\omega}_f \boldsymbol{x}_{t-1}$
$\boldsymbol{v}_f \boldsymbol{h}_{t-2}$
σ
z_{t-1}
p
$f(\cdot)$
$\boldsymbol{v}\boldsymbol{h}_{t-1}$
$\boldsymbol{\omega}\boldsymbol{x}_t$
σ
$\boldsymbol{h}_t$
LBRU cell

Fig. 5. The layer-wise recursion with a forget gate.

which is quite similar to the function of the reset gate in a GRU:

$$\mathbf{n}_t = \tanh(\boldsymbol{\omega}_{in} \boldsymbol{x}_t + \mathbf{b}_{in} + \mathbf{r}_t \odot (\boldsymbol{\omega}_{hn} \boldsymbol{h}_{t-1} + \mathbf{b}_{hn})) \tag{37}$$

Besides the activation function, another main difference is that the forget gate z_{t-1} is computed in the previous time step. If z_{t-1} degrades to a constant 1, we get the formulation of a basic recurrent layer that is used in practice.

V. Backward recursion

The recursions described thus far only yield accurate probabilities at time $t = T$. The earlier ones $(1 < t < T)$ depend upon future observations. This is normally corrected via the backward passes of either the Kalman smoother or forward-backward algorithm. In this section, we derive backward recursions for the recurrent units derived above. In fact, the ability to do this is one of the most compelling reasons to derive probabilistic recurrence.

A. Unit-wise recursion

Although the unit-wise recurrence (without approximations) is unstable, it turns out to be beneficial (see section VII) to derive the backward pass. It can be done without adding extra parameters, making it directly comparable to the GRU.

Following the method for the Kalman smoother, we first integrate over the state at time t and the context variable,

$$P\left(\phi_{t-1,i} \mid \boldsymbol{X}_t\right) = \sum_{\phi_{t,i},\zeta_{t-1}} P\left(\phi_{t-1,i} \mid \phi_{t,i}, \zeta_{t-1}, \boldsymbol{X}_t\right) P\left(\phi_{t,i}, \zeta_{t-1} \mid \boldsymbol{X}_t\right) \tag{38}$$

$$\begin{aligned} &= P\left(\phi_{t-1,i} \mid \phi_{t,i}, \zeta_{t-1}, \boldsymbol{X}_t\right) h_{t,i} z_{t-1} \\ &\quad + P\left(\phi_{t-1,i} \mid \bar{\phi}_{t,i}, \zeta_{t-1} \boldsymbol{X}_t\right)(1-h_{t,i}) z_{t-1} \\ &\quad + P\left(\phi_{t-1,i} \mid \phi_{t,i}, \bar{\zeta}_{t-1}, \boldsymbol{X}_t\right) h_{t,i}(1-z_{t-1}) \\ &\quad + P\left(\phi_{t-1,i} \mid \bar{\phi}_{t,i}, \bar{\zeta}_{t-1} \boldsymbol{X}_t\right)(1-h_{t,i})(1-z_{t-1}). \end{aligned} \tag{39}$$

Note that, given $\phi_{t,i}$, $P\left(\phi_{t-1,i}\right)$ is conditionally independent of any data after time $t-1$. Equations 40–47 show how to use Bayes's theorem to expand the remaining terms. Putting the above together, we initialise

$$h'_{T,i} = h_{T,i} \tag{48}$$

then recurse

$$h'_{t-1} = P\left(\phi_{t-1,i} \mid \boldsymbol{X}_t\right) \tag{49}$$

$$= h'_{t,i} z_{t-1} + h_{t-1,i} h'_{t,i}(1-z_{t-1}) + h_{t-1,i}(1-h'_{t,i})(1-z_{t-1}) \tag{50}$$

$$= (1-z_{t-1}) h_{t-1,i} + z_{t-1} h'_{t,i}. \tag{51}$$

B. Layer-wise recursion

Now we consider the case that $\phi_{t-1,i}$ is dependent on the whole vector $\boldsymbol{\phi}_t$.

$$P\left(\phi_{t-1,i} \mid \boldsymbol{X}_t\right) = \sum_{\boldsymbol{\phi}_t,\zeta_{t-1}} P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \zeta_{t-1}, \boldsymbol{X}_t\right) P\left(\boldsymbol{\phi}_t, \zeta_{t-1} \mid \boldsymbol{X}_t\right) \tag{52}$$

$$= z_{t-1} \sum_{\boldsymbol{\phi}_t} P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \zeta_{t-1}, \boldsymbol{X}_t\right) P\left(\boldsymbol{\phi}_t \mid \boldsymbol{X}_t\right) + (1-z_{t-1}) \sum_{\boldsymbol{\phi}_t} P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \bar{\zeta}_{t-1}, \boldsymbol{X}_t\right) P\left(\boldsymbol{\phi}_t \mid \boldsymbol{X}_t\right) \tag{53}$$

Equations 54–58 show how to use use Bayes's theorem to expand the remaining terms, where $\boldsymbol{\phi}_{t-1,\bar{i}}$ denotes the features of all the units in the layer except the i^{th} unit and p_k is the prior probability of unit k. The first term $P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \zeta_{t-1}, \boldsymbol{X}_t\right)$ seems intractable, although it allows us to re-use the weights learnt from the forward pass to smooth the output via backward recursion. Now suppose there is another binary state variable, ξ_t, where $\xi_t = 1$ indicates the future context remaining relevant, meaning that ϕ_t is dependent on ϕ_{t+1} and $\xi = 0$ indicates that the future context is irrelevant. We can assign a new probability, $s_t = P\left(\xi_t = 1 \mid \boldsymbol{X}_t\right)$ and the inverse $(1-s_t) = P\left(\xi_t = 0 \mid \boldsymbol{X}_t\right)$. We assume ξ_t is independent of future observations $\boldsymbol{X}_{t+1}^T$. Thus, we can write:

$$P\left(\phi_{t-1,i} \mid \boldsymbol{X}_T\right) = \sum_{\boldsymbol{\phi}_t,\xi_t} P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \xi_{t-1}, \boldsymbol{X}_T\right) P\left(\boldsymbol{\phi}_t, \xi_{t-1} \mid \boldsymbol{X}_T\right) \tag{59}$$

$$= s_{t-1} \sum_{\boldsymbol{\phi}_t} P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \xi_{t-1}\right) \prod_k P\left(\phi_{t,k} \mid \boldsymbol{X}_T\right) + (1-s_{t-1}) \sum_{\boldsymbol{\phi}_t} P\left(\phi_{t-1,i} \mid \bar{\xi}_{t-1}, \boldsymbol{X}_{t-1}\right) P\left(\boldsymbol{\phi}_t \mid \boldsymbol{X}_T\right) \tag{60}$$

$$= s_{t-1} \sum_{\boldsymbol{\phi}_t} P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \xi_{t-1}\right) \prod_k P\left(\phi_{t,k} \mid \boldsymbol{X}_T\right) \tag{61}$$

$$+ h_{t-1,i}(1-s_{t-1}). \tag{62}$$

Similarly, we model $P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \xi_{t-1}\right)$ as $\boldsymbol{\omega}_i^{\mathsf{T}} \boldsymbol{\phi}_t$, the product of a trainable vector $\boldsymbol{\omega}_i$ and $\boldsymbol{\phi}_t$, and denote $h'_{t,i} = P\left(\phi_{t,i} \mid \boldsymbol{X}_T\right)$ and put the above together, we initialise

$$h'_{T,i} = h_{T,i} \tag{63}$$

then recurse

$$P\left(\phi_{t-1,i} \mid \boldsymbol{X}_T\right) = h'_{t-1,i} = s_t(\boldsymbol{\omega}_i^{\mathsf{T}} \boldsymbol{h}'_t + c) + h_{t-1,i}(1-s_t), \tag{64}$$

where $c = \sum_{j \in \{j \mid \omega_{j,i} < 0\}} \omega_{j,i}$. It is sensible to apply the same constraints discussed in Section IV-C to the backward recurrent matrix and add the bias term.

The layer-wise backward pass hence requires extra parameters. In this sense it is not directly comparable to a similar GRU. Nevertheless, the parameter count is smaller than for a bidirectional GRU. The repercussions of this are examined in section VII.

VI. Probabilistic input

In examining the probabilistic forget derivations above, whilst we set out to formalise the CEC of the LSTM, the result is closer to the reset gate of a GRU. In this section, we show that the update gate of a GRU can also be derived rather simply.

A. Recursion

In the same spirit as the previous section, say there is a binary state variable, ρ, where $\rho = 1$ indicates the current input is relevant, and $\rho = 0$ indicates that it is not relevant. We can assign a probability, $r_t = P\left(\rho_t = 1 \mid \boldsymbol{X}_t\right)$ and the inverse $(1-r_t) = P\left(\rho_t = 0 \mid \boldsymbol{X}_t\right)$. We assume if the current input is irrelevant, then ϕ_t is completely dependent on ϕ_{t-1}. For a given feature, ϕ_i, the derivation is shown in equations 65–69 below. The first term follows the same derivations in previous sections. This is illustrated in Fig. 6, where, as before, the unit-wise and layer-wise recursions are merged, and the gate recursion remains ad-hoc; this provides for a fair comparison with GRU in section VII.

This correlates to the update function in a GRU:

$$\boldsymbol{h}_t = (1-\mathbf{z}_t) \odot \mathbf{n}_t + \mathbf{z}_t \boldsymbol{h}_{(t-1)}, \tag{70}$$

$$P\left(\phi_{t-1,i} \mid \phi_{t,i}, \zeta_{t-1}, \boldsymbol{X}_t\right) = \frac{P\left(\phi_{t,i} \mid \phi_{t-1,i}, \zeta_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)}{\sum_{\phi_{t-1,i}} P\left(\phi_{t,i} \mid \phi_{t-1,i}, \zeta_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)} \tag{40}$$

$$= \frac{1 \times h_{t-1,i}}{1 \times h_{t-1,i} + 0 \times (1 - h_{t-1,i})} = 1. \tag{41}$$

$$P\left(\phi_{t-1,i} \mid \bar{\phi}_{t,i}, \zeta_{t-1}, \boldsymbol{X}_t\right) = \frac{P\left(\bar{\phi}_{t,i} \mid \phi_{t-1,i}, \zeta_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)}{\sum_{\phi_{t-1,i}} P\left(\bar{\phi}_{t,i} \mid \phi_{t-1,i}, \zeta_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)} \tag{42}$$

$$= \frac{0 \times h_{t-1,i}}{0 \times h_{t-1,i} + 1 \times (1 - h_{t-1,i})} = 0. \tag{43}$$

$$P\left(\phi_{t-1,i} \mid \phi_{t,i}, \bar{\zeta}_{t-1}, \boldsymbol{X}_t\right) = \frac{P\left(\phi_{t,i} \mid \phi_{t-1,i}, \bar{\zeta}_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)}{\sum_{\phi_{t-1,i}} P\left(\phi_{t,i} \mid \phi_{t-1,i}, \bar{\zeta}_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)} \tag{44}$$

$$= \frac{p_i h_{t-1,i}}{p_i h_{t-1,i} + p_i(1 - h_{t-1,i})} = h_{t-1,i}. \tag{45}$$

$$P\left(\phi_{t-1,i} \mid \bar{\phi}_{t,i}, \bar{\zeta}_{t-1}, \boldsymbol{X}_t\right) = \frac{P\left(\bar{\phi}_{t,i} \mid \phi_{t-1,i}, \bar{\zeta}_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)}{\sum_{\phi_{t-1,i}} P\left(\bar{\phi}_{t,i} \mid \phi_{t-1,i}, \bar{\zeta}_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)} \tag{46}$$

$$= \frac{(1 - p_i) h_{t-1,i}}{(1 - p_i) h_{t-1,i} + (1 - p_i)(1 - h_{t-1,i})} = h_{t-1,i}. \tag{47}$$

$$P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \zeta_{t-1}, \boldsymbol{X}_t\right) = \frac{P\left(\boldsymbol{\phi}_t \mid \phi_{t-1,i}, \zeta_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)}{\sum_{\boldsymbol{\phi}_{t-1}} P\left(\boldsymbol{\phi}_t \mid \boldsymbol{\phi}_{t-1}, \zeta_{t-1}\right) P\left(\boldsymbol{\phi}_{t-1} \mid \boldsymbol{X}_{t-1}\right)} \tag{54}$$

$$= \frac{\sum_{\boldsymbol{\phi}_{t-1,\bar{i}}} P\left(\boldsymbol{\phi}_t \mid \phi_{t-1,i}, \boldsymbol{\phi}_{t-1,\bar{i}}, \zeta_{t-1}\right) P\left(\boldsymbol{\phi}_{t-1,\bar{i}} \mid \boldsymbol{X}_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)}{\sum_{\boldsymbol{\phi}_{t-1}} P\left(\boldsymbol{\phi}_t \mid \boldsymbol{\phi}_{t-1}, \zeta_{t-1}\right) P\left(\boldsymbol{\phi}_{t-1} \mid \boldsymbol{X}_{t-1}\right)} \tag{55}$$

$$P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \bar{\zeta}_{t-1}, \boldsymbol{X}_t\right) = \frac{\sum_{\boldsymbol{\phi}_{t-1,\bar{i}}} P\left(\boldsymbol{\phi}_t \mid \phi_{t-1,i}, \boldsymbol{\phi}_{t-1,\bar{i}}, \bar{\zeta}_{t-1}\right) P\left(\boldsymbol{\phi}_{t-1,\bar{i}} \mid \boldsymbol{X}_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)}{\sum_{\boldsymbol{\phi}_{t-1}} P\left(\boldsymbol{\phi}_t \mid \boldsymbol{\phi}_{t-1}, \bar{\zeta}_{t-1}\right) P\left(\boldsymbol{\phi}_{t-1} \mid \boldsymbol{X}_{t-1}\right)} \tag{56}$$

$$= \frac{\prod_k p_k}{\prod_k p_k (1 + (1 - h_{t-1,i})/h_{t-1,i})} \tag{57}$$

$$= h_{t-1,i} \tag{58}$$

---

where n_t is defined as equation 37 and z_t is the update gate computed as

$$\mathbf{z}_t = \sigma(\boldsymbol{\omega}_{iz}\boldsymbol{x}_t + \mathbf{b}_{iz} + \boldsymbol{\omega}_{hz}\boldsymbol{h}_{(t-1)} + \mathbf{b}_{hz}) \tag{71}$$

It may be argued that the input gate and the forget gate have simliar functionality. Indeed, if we only keep the forget gate and let $z_t = P\left(\rho_t = 0 \mid \boldsymbol{X}_t\right)$, this leads to the MGU [12]; If we keep the forget gate always equal to 1, it leads to the Li-GRU [14].

We do not derive a backward recursion for the input gate. Rather, the resulting resemblance to the GRU provides us with a candidate architecture to compare experimentally; this is reported in section VII.

B. Summary

The forward pass of this final BRU can be summarised as:

$$\mathbf{z}_t = \sigma(\boldsymbol{\omega}_{iz}\boldsymbol{x}_t + \boldsymbol{\omega}_{hz}\boldsymbol{h}_{t-1} + \mathbf{b}_z) \tag{72}$$

$$\mathbf{r}_t = \sigma(\boldsymbol{\omega}_{ir}\boldsymbol{x}_t + \boldsymbol{\omega}_{hr}\boldsymbol{h}_{t-1} + \mathbf{b}_r) \tag{73}$$

$$\mathbf{n}_t = \sigma(\boldsymbol{\omega}_{ih}\boldsymbol{x}_t + \mathbf{b}_{ih} + \mathbf{z}_{t-1} \odot (\boldsymbol{\omega}_{hh}\boldsymbol{h}_{t-1} + \mathbf{b}_{hh})) \tag{74}$$

$$\boldsymbol{h}_t = (1 - \mathbf{r}_t) \odot \mathbf{n}_t + \mathbf{r}_t \odot \boldsymbol{h}_{t-1}, \tag{75}$$

In the backward pass, two cases can be considered, namely unit-wise BRU (UBRU):

$$\boldsymbol{h}'_{t-1} = \boldsymbol{h}'_t \odot \mathbf{z}_t + \boldsymbol{h}_{t-1} \odot (1 - \mathbf{z}_t) \tag{76}$$

and layer-wise BRU (LBRU):

$$\mathbf{s}_t = \sigma(\boldsymbol{\omega}_{is}\boldsymbol{x}_t + \mathbf{b}_{is} + \boldsymbol{\omega}_{hs}\boldsymbol{h}_{t-1} + \mathbf{b}_{hs}) \tag{77}$$

$$\boldsymbol{h}'_{t-1} = (\boldsymbol{\omega}_{hhb}\boldsymbol{h}'_t + \mathbf{b}_{hhb}) \odot \mathbf{s}_t + \boldsymbol{h}_{t-1} \odot (1 - \mathbf{s}_t). \tag{78}$$

Note that in the above, we retain the ad-hoc gate recurrence as we find that it performs marginally better than not doing so. However, there is currently no probabilistic reason to do

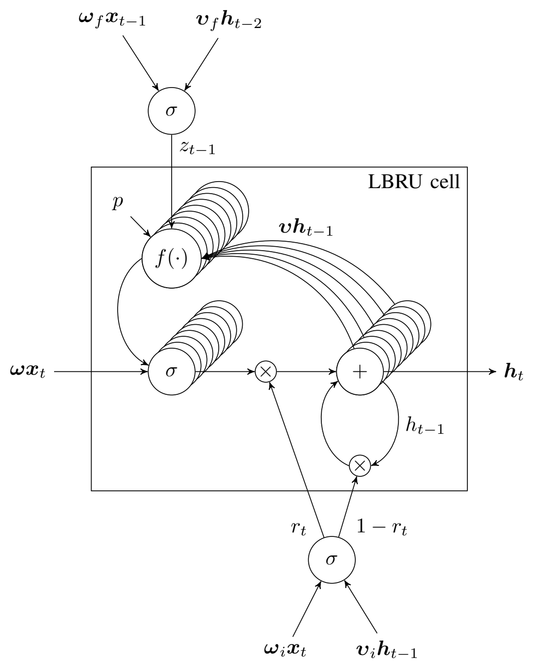
$$h_{t,i} = P\left(\phi_{t,i} \mid \boldsymbol{X}_t\right) \tag{65}$$

$$= \sum_{\rho_{t,i}} P\left(\phi_{t,i} \mid \boldsymbol{X}_t, \rho_{t,i}\right) P\left(\rho_{t,i} \mid \boldsymbol{X}_t\right) \tag{66}$$

$$= P\left(\phi_{t,i} \mid \boldsymbol{X}_t, \rho_{t,i}\right) P\left(\rho_{t,i} \mid \boldsymbol{X}_t\right) + \sum_{\phi_{t-1,i}} P\left(\phi_{t,i} \mid \boldsymbol{X}_t, \phi_{t-1,i}, \bar{\rho}_{t,i}\right) P\left(\bar{\rho}_{t,i} \mid \boldsymbol{X}_t\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_t\right) \tag{67}$$

$$\approx P\left(\phi_{t,i} \mid \boldsymbol{X}_t\right) P\left(\rho_{t,i} \mid \boldsymbol{X}_t\right) + \sum_{\phi_{t-1,i}} P\left(\phi_{t,i} \mid \phi_{t-1,i}, \bar{\rho}_{t,i}\right) P\left(\bar{\rho}_{t,i} \mid \boldsymbol{X}_t\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right) \tag{68}$$

$$= r_{t,i} P\left(\phi_{t,i} \mid \boldsymbol{X}_t\right) + (1 - r_{t,i}) h_{t-1,i} \tag{69}$$



$\boldsymbol{\omega}_f \boldsymbol{x}_{t-1}$
$\boldsymbol{v}_f \boldsymbol{h}_{t-2}$
σ
z_{t-1}
LBRU cell
p
$f(\cdot)$
$\boldsymbol{v}\boldsymbol{h}_{t-1}$
$\boldsymbol{\omega}\boldsymbol{x}_t$
σ
$+$
$\boldsymbol{h}_t$
h_{t-1}
r_t
$1 - r_t$
σ
$\boldsymbol{\omega}_i \boldsymbol{x}_t$
$\boldsymbol{v}_i \boldsymbol{h}_{t-1}$

Fig. 6. The layer-wise recursion with a forget gate and an input gate.

so. We set this matter aside for the future. With reference to section IV, in defining the gates as vectors, we are assuming one gate per feature; this is usual in LSTM and GRU, but not a constraint.

VII. Experiments

We present evaluations of the techniques described thus far on automatic speech recognition (ASR) tasks. Recurrent networks are particularly suited to ASR as there is an explicit time dimension and well known context dependency. Reciprocally, ASR is a difficult task that has driven recent advances in deep learning [27]–[30].

A. Hypotheses

In running experiments, we are testing the Bayesian recurrent unit (BRU) derived in the previous three sections. This raises two explicit hypotheses:

1) We would expect the incorporation of a backward pass to improve upon the performance of a (forward-only) GRU.
2) We would expect the LBRU to approach the performance of a conventional GRU-based BiRNN architecture. It has the same contextual knowledge, but does not have higher representational capability. If it falls short of a BiRNN architecture then either the approximations in the derivation are not valid, or the BiRNN is taking advantage of temporal asymmetry in the data.

This is all dependent upon the number of parameters: A BiRNN has roughly twice as many parameters as a Bayesian RNN with a backward pass.

B. Corpora and method

Detailed statistics of the corpora considered in this work are summarised in table I.

TABLE I
Statistics of datasets used in this work: speakers and sentences are counts, the amounts of speech data for training and evaluation sets are in hours.

Dataset	Speakers	Sentences	Train	Eval
TIMIT	462	3696	5	0.16
WSJ	283	37416	81.3	0.7
AMI-IHM	10487	98397	70.3	8.6

A first set experiments with the TIMIT corpus [**?**] was performed to test the proposed model for a phoneme recognition task. We used the standard 462-speaker training set and removed all SA records, since they may bias the results. A separate development set of 50 speakers was used for tuning all meta-parameters including the learning schedule and multiple learning rates. Results are reported using the 24-speaker core test set, which has no overlap with the development set. Following the implementation of [**?**], [14], all the recurrent networks tested on this dataset have 5 layers, each consisting 550 units in each direction and use 40 fMLLR features (extracted based on the Kaldi recipe) as the input.

The second set of experiments was carried out on the Wall Street Journal (WSJ) speech corpus to gauge the suitability of the proposed model for large vocabulary speech recognition. We used the standard configuration si284 dataset for training, dev93 for tuning hyper-parameters, and eval92 for evaluation. All the tested recurrent networks have 3 layers, each consisting of 320 units in each direction. We used 40 fMLLR features as input for speaker adaptation.

The TIMIT and WSJ datasets yield results with modest statistical significance. In order to yield more persuasive significance, a set of experiments was also conducted on the AMI corpus [33] with the data recorded through individual headset microphones (IHM). The AMI corpus contains recordings of spontaneous conversations in meeting scenarios, with 70 hours of training data, 9 hours of development, and 8 hours of test data. All the tested recurrent networks have 3 layers, each consisting of 512 units in each direction and use 40 fMLLR features as the input.

The neural networks were trained to predict context-dependent phone targets. The labels were derived by performing a forced alignment procedure on the training set using GMM-HMM, as in the standard recipe of Kaldi[1] [34]. During testing, the posterior probabilities generated for each frame by the neural networks are normalised by their priors, then processed by an HMM-based decoder, which estimates the sequence of words by integrating the acoustic, lexicon and language model information. The neural networks of the ASR system were implemented in PyTorch[2], including, crucially, the gradient calculation; they were coupled with the Kaldi decoder [34] to form a context-dependent RNN-HMM speech recogniser.

C. Training details

The network architecture adopted for the experiments contains multiple recurrent layers, which are stacked together prior to the final softmax context-dependent (senon) classifier. If the networks are bidirectional, the forward hidden states and the backward hidden states at each layer are concatenated before feeding to the next layer. A dropout rate of 0.2 was used for regularisation. Moreover, batch normalization [**?**] was adopted on each layer to accelerate the training. The optimization was performed using the Adaptive Moment Estimation (Adam) algorithm [36] running for 24 epochs with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$. The performance on the cross validation set was monitored after each epoch, while the learning rate was halved when the performance improvement dropped below a certain threshold (0.001).

D. Phoneme recognition performance on TIMIT

In order to confirm the suitability of the proposed model for acoustic modeling, TIMIT was first considered to reduce the linguistic effects (such as lexicon and language model) on the performance evaluation. The state of the art for this task is probably that of Ravanelli et al. [14], with a phone error rate (PER) of 14.9%. We duplicate the architecture of those authors and aim for a similar figure. We performed the comparison with GRU as shown in Fig. 7. The error bars indicate equal-tailed 95% credible interval for a beta assumption for the error rate. The numbers in the parentheses indicate the number of parameters each model contains. It is clear that the unidirectional GRU (Uni-GRU) is significantly worse than bidirectional GRU (Bi-GRU) as the credible intervals do not overlap. By contrast, the unit-wise BRU (UBRU) yields much better performance compared to Uni-GRU with exactly the same model size, and the layer-wise BRU (LBRU) is slightly better than UGRU, yielding similar performance to Bi-GRU.

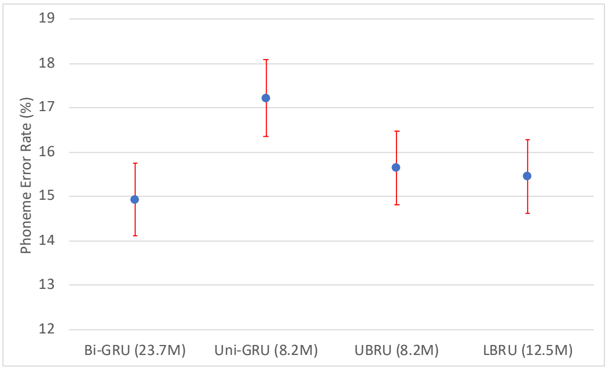


Fig. 7. Phoneme Error Rate (%) on TIMIT for various RNN architectures.

Since the test set in TIMIT is quite small, we also performed a matched-pair t-test between Uni-GRU and UBRU, the test statistic being the utterance-wise difference in word-level errors normalised by the reference length. This yields $p < 0.001$, showing that the UBRU is significantly better. This confirms our first hypothesis that the incorporation of a backward pass can improve upon the performance of a unidirectional GRU. The t-test between Bi-GRU and LBRU yields $p = 0.230$, which implies there is no significant difference between the two systems. The two comparisons together show that our proposed model can achieve performance indistinguishable from the Bi-GRU, without the explicit extra backward recurrence.

Although the difference between Bi-GRU and LBRU is not significant, the latter one is slightly worse. This can be explained by our second hypothesis. Physiological filters are known to have asymmetric impulse responses [37]. This is one explanation for the large improvement arising from doubling up the Uni-GRU to explicitly modelling the backward recursion. However, the proposed BRU does not have the explicit extra backward recurrence of the BiRNN architectures. Therefore, we further doubled up the LBRU to be explicitly bidirectional and compared it with Bi-GRU and Bi-LSTM, as shown in Fig. 8. Similarly, we plot the error bars and the sizes of the models; this shows that GRU and LSTM perform almost the same while the Bi-LBRU seems to be slightly better with a few more parameters, although the difference is insignificant from the t-test ($p = 0.43$). Our hypothesis is that BRU has a stronger modelling ability in each of the directions because the prediction is always conditioned on the whole sequence due to the implicit backward recursion. We note that the average PER of 14.6% obtained with Bi-LBRU outperforms the state of the art 14.9% of [14] on the TIMIT test-set, although it is well within the 95% confidence bounds.

E. Speech recognition performance on WSJ

Since TIMIT is too small to yield significant comparisons, in this sub-section, we evaluate the RNNs on WSJ, a large vocabulary continuous speech recognition task. Following the

[1]http://kaldi-asr.org/
[2]https://pytorch.org/

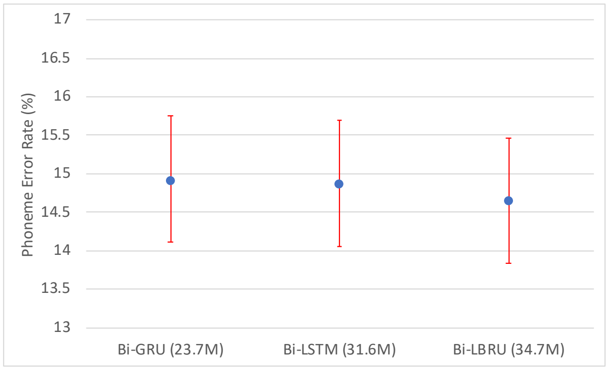


Fig. 8. Phoneme Error Rate (%) on TIMIT for various RNN architectures.

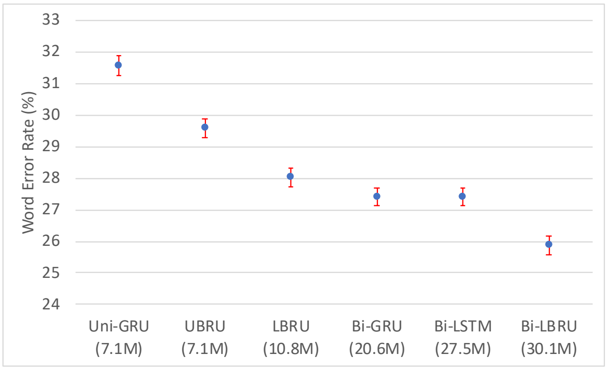


Fig. 10. Word Error Rate (%) on AMI for various RNN architectures.

TIMIT case, we plot the word error rate (WER) in Fig. 9, together with the corresponding error bars and model sizes. These results exhibit a similar trend to that observed on TIMIT. Both UBRU and LBRU outperform the Uni-GRU ($p = 0.19$ from the t-test). LBRU is slightly better than UBRU and it yields very similar performance to that of Bi-GRU ($p = 0.21$ from the t-test). The Bi-LBRU still performs slightly better than Bi-GRU and Bi-LSTM. Again, the differences are not significant owing to the fact that the test set of WSJ is still quite small. Overall, the results are comparable with the baselines reported in the Kaldi software; for instance, 4.27% using a Bi-LSTM and i-vectors.

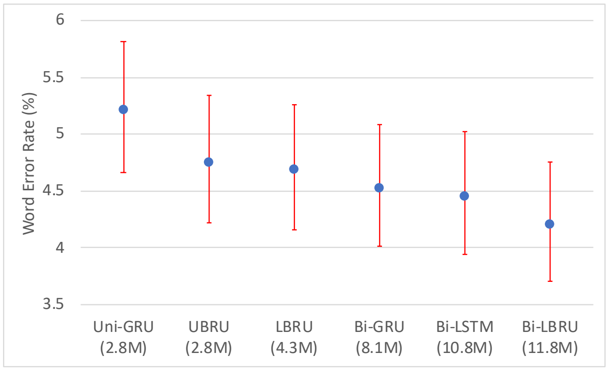


Fig. 9. Word Error Rate (%) on WSJ for various RNN architectures.

F. Speech recognition performance on AMI

Owing to the small test set of WSJ, in this sub-section we conduct the evaluation on AMI, which is a more challenging task with a much larger test set. AMI is more challenging as the data is recorded in meetings, capturing natural spontaneous conversations between participants who play different roles in the meeting. Overlapping speech segments appear in both training and testing. State of the art results on AMI tend to be for complicated systems with elements of speaker and environment adaptation, e.g., Kanda et al [38] report a WER of 17.84%. Rather than aim to duplicate such results, we simply aim for a self-consistent comparison of techniques; our results are in the same range as the 26.8% of Dighe et al [39].

Fig. 10 summarises the results obtained on AMI. These results show the same trend as previous experiments, but also exhibit more significant differences. Both UBRU and LBRU significantly outperform Uni-GRU while LBRU is also significantly better than UBRU ($p < 0.001$ from the t-test), showing that the layer-wise backward recursion is able to capture richer characteristics in the backward transition. Comparison between LBRU and Bi-GRU shows that LBRU can achieve similar performance without an extra explicit backward network. Bi-LSTM does not have any advantages over Bi-GRU, although it contains one more gate and, therefore, more parameters. However, if we double up the LBRU to be explicitly bidirectional, the model yields significantly better performance than both Bi-GRU and Bi-LSTM ($p < 0.001$ from the t-test). This confirms the hypothesis that BRU has a stronger unidirectional modelling ability and explicit bidirectional modelling can help capture the asymmetric characteristics in physiological filters.

VIII. Conclusion

Given a probabilistic interpretation of common neural network components, it is possible to derive recurrent components in the same spirit. Such components have two advantages:

1) The architecture of the recursion is dictated by the probabilistic formulation, removing otherwise ad-hoc choices.
2) They naturally support a backward recursion of the type used in Kalman smoothers and the forward-backward algorithm of the HMM.

Unit-wise recursions follow analytically, but are found to lead to instabilities. Approximations lead to stable layer-wise recursions. Nevertheless, useful backward recursions can be derived for both cases. The resulting Bayesian recurrent unit (BRU) can be configured with a probabilistic input gate, being directly comparable to a common GRU.

Evaluation on simple and on state of the art speech recognition tasks shows that:

1) Even the unit-wise backward recursion can out-perform a standard GRU.
2) A more involved layer-wise backward recursion can approach the performance of a bidirectional GRU. This shows that the approximations in the derivations are reasonable.

Further, an explicit bidirectional BRU can out-perform a state of the art bidirectional GRU.

There are some ad-hoc methods in our approach: the gate recurrences are retained for performance; some approximations may be better formulated. These remain matters for future research. Nevertheless, we have shown that recurrence in neural networks can be formulated much more rigorously than conventional wisdom would hold. This in turn can lead to significant performance advantages.

Acknowledgment

We are grateful to our colleagues Apoorv Vyas and Bastian Schnell for battling through an early manuscript and providing comments that rendered it more accessible. We also extend our thanks to the editor and two referees for their insights.

Part of this work was conducted within the scope of the Research and Innovation Action SUMMA, which has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688139.

References

[1] R. E. Kalman, "A new approach to linear filtering and prediction problems," *ASME Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, Mar. 1960.

[2] L. L. Scharf, *Statistical Signal Processing. Detection, Estimation and Time Series Analysis*. Addison Wesley, 1991.

[3] L. E. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state Markov chains," *The Annals of Mathematical Statistics*, vol. 37, no. 6, pp. 1554–1563, December 1966.

[4] L. R. Bahl, F. Jelinek, and R. L. Mercer, "A maximum likelihood approach to continuous speech recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-5, no. 2, pp. 179–190, March 1983.

[5] D. E. Rumelhart and J. L. McClelland, *Parallel Distributed Processing. Explorations in the Microstructure of Cognition*. MIT Press, July 1986, vol. 1: Foundations.

[6] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, October 1986.

[7] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Computation*, vol. 1, no. 2, pp. 270–280, 1989.

[8] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, November 1997.

[9] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," *Neural Computation*, vol. 12, pp. 2451–2471, 2000.

[10] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with LSTM recurrent networks," *Journal of Machine Learning Research*, vol. 3, pp. 115–143, August 2002.

[11] K. Cho, B. van Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, October 2014, pp. 1724–1734.

[12] G.-B. Zhou, J. Wu, C.-L. Zhang, and Z.-H. Zhou, "Minimal gated unit for recurrent neural networks," *International Journal of Automation and Computing*, vol. 13, no. 3, pp. 226–234, June 2016.

[13] M. Ravanelli, P. Brakel, M. Omologo, and Y. Bengio, "Improving speech recognition by revising gated recurrent units," in *Proceedings of Interspeech*, Stockholm, Sweden, August 2017, pp. 1308–1312.

[14] ——, "Light gated recurrent units for speech recognition," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 2, pp. 92–102, Apr. 2018.

[15] H. Bourlard and C. J. Wellekens, "Links between Markov models and multilayer perceptrons," in *Advances in Neural Information Processing Systems 1*, D. S. Touretzky, Ed. Morgan Kaufmann, 1989, pp. 502–510.

[16] J. S. Bridle, "Alpha-nets: A recurrent 'neural' network architecture with a hidden Markov model interpretation," *Speech Communication*, vol. 9, no. 1, Feb. 1990.

[17] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997.

[18] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM networks," in *Proceedings of the 2005 IEEE International Joint Conference on Neural Networks*, Montreal, Quebec, Canada, July 2005.

[19] J. S. Bridle, "Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition," in *Neurocomputing*, ser. NATO ASI Series F: Computer and Systems Sciences, F. Fogelman Soulié and J. Hérault, Eds. Berlin Heidelberg: Springer-Verlag, 1990, vol. 68, pp. 227–236.

[20] M. D. Richard and R. P. Lippman, "Neural network classifiers estimate Bayesian a posteriori probabilities," *Neural Computation*, vol. 3, no. 4, pp. 461–483, Winter 1991.

[21] D. J. C. MacKay, "Bayesian interpolation," *Neural Computation*, vol. 4, no. 3, pp. 415–447, May 1992.

[22] ——, "A practical Bayesian framework for backpropagation networks," *Neural Computation*, vol. 4, no. 3, pp. 448–472, May 1992.

[23] ——, "The evidence framework applied to classification networks," *Neural Computation*, vol. 4, no. 5, pp. 720–736, May 1992.

[24] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[25] C. Dugas, Y. Bengio, F. Bélisle, C. Nadeau, and R. Garcia, "Incorporating second-order functional knowledge for better option pricing," in *Advances in Neural Information Processing Systems 13*, T. K. Leen, T. G. Dietterich, and V. Tresp, Eds. MIT Press, 2001, pp. 472–478.

[26] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, Fort Lauderdale, FL, USA, 2011, pp. 315–323.

[27] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *ICML '06 Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, USA, June 2006, pp. 369–376.

[28] F. Seide, G. Li, and D. Yu, "Conversational speech transcription using context-dependent deep neural networks," in *Proceedings of Interspeech*, Florence, Italy, August 2011, pp. 437–440.

[29] W. Xiong, J. Droppo, X. Huang, F. Seide, M. Seltzer, A. Stolcke, D. Yu, and G. Zweig, "The Microsoft 2016 conversational speech recognition system," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, New Orleans, LA, USA, March 2017, pp. 5255–5259.

[30] H. Hadian, H. Sameti, D. Povey, and S. Khudanpur, "End-to-end speech recognition using lattice-free MMI," in *Proceedings of Interspeech*, Hyderabad, India, September 2018, pp. 12–16.

[31] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, and D. S. Pallett, "DARPA TIMIT acoustic-phonetic continous speech corpus CD-ROM. NIST speech disc 1-1.1," *NASA STI/Recon technical report n*, vol. 93, 1993.

[32] M. Ravanelli, T. Parcollet, and Y. Bengio, "The pytorch-kaldi speech recognition toolkit," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2019.

[33] J. Carletta, S. Ashby, S. Bourban, M. Guillemot, M. Kronenthal, G. Lathoud, M. Lincoln, I. McCowan, T. Hain, W. Kraaij, W. Post, J. Kadlec, P. Wellner, M. Flynn, and D. Reidsma, "The AMI meeting corpus," in *Proceedings of MLMI'05*, Edinburgh, 2005.

[34] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, Hawaii, USA, December 2011, pp. 1–4.

[35] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv:1412.6980, Dec. 2014, published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015. [Online]. Available: https://arxiv.org/abs/1412.6980

[37] P.-E. Honnet, B. Gerazov, A. Gjoreski, and P. N. Garner, "Intonation modelling using a muscle model and perceptually weighted matching pursuit," *Speech Communication*, vol. 97, pp. 81–93, March 2018.

[38] N. Kanda, Y. Fujita, and K. Nagamatsu, “Lattice-free state-level minimum Bayes risk training of acoustic models,” in *Proceedings of Interspeech*, Hyderabad, India, Sep. 2018.

[39] P. Dighe, H. Bourlard, and A. Asaei, “Far-field ASR using low-rank and sparse soft targets from parallel data,” in *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, Athens, Greece, Dec. 2018, pp. 581–587.