Appendices

A Proof of Proposition 11

Proof. Let's consider a single-layer QRNN with 2-window convolutions:

$$\begin{aligned}
\mathbf{f}_t &= \boldsymbol{\sigma}\left(\mathbf{V}_f\mathbf{v}_{t-1} + \mathbf{W}_f\mathbf{v}_t + \mathbf{b}_f\right),\\
\mathbf{u}_t &= (\mathbf{1} - \mathbf{f}_t) \odot \boldsymbol{g}\big(\mathbf{V}_u\mathbf{v}_{t-1} + \mathbf{W}_u\mathbf{v}_t + \mathbf{b}_u\big),\\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{u}_t.
\end{aligned}$$

A similar analysis applies to T-GRUs and T-LSTMs directly, and it should be straightforward to generalize the discussion to QRNNs with larger convolution windows.

Let Σ denote the alphabet, and let $\mathbf{x} = x_1x_2\dots x_n \in \Sigma^+$ be a nonempty input string. consider a WFSA over the real semiring with $2\,|\Sigma| + 1$ states, where q_0 is the initial state with $\lambda(q_0) = 1$; $|\Sigma|$ of them are final states $\mathcal{Q}_2 = \{q_\alpha\}_{\alpha\in\Sigma}$, with $\rho(q_\alpha) = 1$, and the remaining $|\Sigma|$ states are denoted by $\mathcal{Q}_1 = \{p_\alpha\}_{\alpha\in\Sigma}$.

The transition weights τ are constructed by

$$\begin{aligned}
&\tau(q_0, p_\alpha, \alpha) = 1, && \forall p_\alpha, \in \mathcal{Q}_1;\\
&\tau(p_\alpha, p_\beta, \beta) = 1, && \forall p_\alpha, p_\beta \in \mathcal{Q}_1;\\
&\tau(p_\alpha, q_\beta, \beta) = \mu_\alpha(\beta), && \forall p_\alpha \in \mathcal{Q}_1, \forall q_\beta \in \mathcal{Q}_2;\\
&\tau(q_\alpha, q_\beta, \beta) = \phi_\alpha(\beta), && \forall q_\alpha, q_\beta \in \mathcal{Q}_2.
\end{aligned}$$

$\tau = 0$ otherwise. Then one dimension of the reccurent updates of a 2-window QRNN is recovered by parameterizing the weight functions as

$$\mu_{x_{t-1}}(x_t) = [\mathbf{u}_t]_i, \quad \phi_{x_{t-1}}(x_t) = [\mathbf{f}_t]_i. \tag{22}$$

The recurrent computation of a 2-window QRNN of hidden size d can then be recovered by collecting d such WFSAs. $\square$

B Proof of Proposition 12

Proof. We present the construction of WFSAs for a single layer n-gram RCNNs of hidden size d.

Let's assume a given input sequence $\mathbf{x} \in \Sigma^+$, with $|\mathbf{x}| > n$, since otherwise one only needs include paddings, just as in a RCNN. Consider a WFSA with $n+1$ states $\mathcal{Q} = \{q_i\}_{i=0}^{n}$ over the real semiring. Use q_0 as the initial state with $\lambda(q_0) = 1$, and q_n as the final state with $\rho(q_n) = 1$. The transition weight function is defined by

$$\tau(q_i, q_j, \alpha) = \begin{cases} 1, & i = j = 0,\\ \phi(\alpha), & j = i > 0,\\ \mu_j(\alpha), & j = i + 1,\\ 0, & \text{otherwise.}\end{cases}$$

Let $z_t^{(j)}$ denote the total score of all paths landing in state q_j *just* after consuming x_t. Let $z_t^{(j)} = 0, j = 0, \dots, n$. By the forward algorithm

$$\begin{aligned}
&z_t^{(0)} = 1,\\
&z_t^{(j)} = z_{t-1}^{(j)}\,\phi(x_t) + z_{t-1}^{(j-1)}\,\mu_j(x_t), \quad j \geq 1.
\end{aligned}$$

Applying similar parametrization to that in §4.2, $z_t^{(n)}$ recovers one dimension of the recurrence. Collecting d such WFSAs we recover the recurrence of a single layer n-gram RCNNs, with $\boldsymbol{\lambda}_t$ being a constant, or depending only on x_t. $\square$

C Proof of Proposition 13

Proof. Closely following the 2-dimensional case in §4.3, let's discuss a single layer ISAN of hidden size d.

Consider a WFSA over the real semiring with $2d$ states. Let d of them, denoted by $\mathcal{Q}_2 = \{q_i\}_{i=d+1}^{2d}$ be the initial states, with $\lambda(q_i) = 1, i = 1, \dots, d$. Denote the other half $\{q_i\}_{i=1}^{d}$ by $\mathcal{Q}_1$. Define transition weight τ by:

$$\tau(q_i, q_j, \alpha) = \begin{cases} 1, & i = j,\ q_i \in \mathcal{Q}_2,\\ \eta_j(\alpha), & i = j + d,\\ \mu_{j,i}(\alpha), & q_i, q_j \in \mathcal{Q}_1,\\ 0, & \text{otherwise.}\end{cases}$$

$\forall \alpha \in \Sigma$.

Using $q_i \in \mathcal{Q}_1$ as the final state with $\rho(q_i) = 1$, and denote the resulting WFSA by $\mathcal{G}_i$. By Forward algorithm, $\mathcal{G}_i$ recovers the ith dimension of the single layer ISAN by letting $[\mathbf{W}_{x_t}]_{i,j} = \mu_{i,j}(x_t)$, and $[\mathbf{b}_{x_t}]_i = \eta_i(x_t)$; the d-dimensional recurrent computation is recovered by a set of WFSAs $\{\mathcal{G}_i\}_{i=1}^{d}$ constructed similarly. $\square$

D Compared Models

This section formally describes the models compared in the experiments (§6.1).

RRNN($\mathcal{B}$). RRNN($\mathcal{B}$) is derived from $\mathcal{B}$ (§4.1).

$$\begin{aligned}
\mathbf{f}_t &= \boldsymbol{\sigma}\big(\mathbf{W}_f\mathbf{v}_t + \mathbf{b}_f\big), && \text{(23a)}\\
\mathbf{u}_t &= (1 - \mathbf{f}_t) \odot \mathbf{W}_u\mathbf{v}_t, && \text{(23b)}\\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{u}_t, && \text{(23c)}\\
\mathbf{o}_t &= \boldsymbol{\sigma}\big(\mathbf{W}_o\mathbf{v}_t + \mathbf{b}_o\big), && \text{(23d)}\\
\mathbf{h}_t &= \tanh(\mathbf{o}_t \odot \mathbf{c}_t). && \text{(23e)}
\end{aligned}$$

RRNN($\mathcal{B}$)$_{\mathbf{m+}}$. Also derived from $\mathcal{B}$, but uses the max-plus semiring (§5.2).

$$\mathbf{f}_t = \log \boldsymbol{\sigma}\big(\mathbf{W}_f \mathbf{v}_t + \mathbf{b}_f\big), \tag{24a}$$
$$\mathbf{u}_t = \mathbf{W}_u \mathbf{v}_t, \tag{24b}$$
$$\mathbf{c}_t = \max\{\mathbf{f}_t + \mathbf{c}_{t-1}, \mathbf{u}_t\}, \tag{24c}$$
$$\mathbf{o}_t = \log \boldsymbol{\sigma}\big(\mathbf{W}_o \mathbf{v}_t + \mathbf{b}_o\big), \tag{24d}$$
$$\mathbf{h}_t = \tanh(\mathbf{o}_t + \mathbf{c}_t). \tag{24e}$$

RRNN($\mathcal{C}$). RRNN($\mathcal{C}$) is derived from $\mathcal{C}$ (§4.2):

$$\mathbf{f}_t^{(j)} = \boldsymbol{\sigma}\big(\mathbf{W}_f^{(j)} \mathbf{v}_t + \mathbf{b}_f^{(j)}\big), \quad j = 1, 2, \tag{25a}$$
$$\mathbf{u}_t^{(j)} = (1 - \mathbf{f}_t^{(j)}) \odot \mathbf{W}_u^{(j)} \mathbf{v}_t, \quad j = 1, 2, \tag{25b}$$
$$\mathbf{c}_t^{(1)} = \mathbf{c}_{t-1}^{(1)} \odot \mathbf{f}_t^{(1)} + \mathbf{u}_t^{(1)}, \tag{25c}$$
$$\mathbf{c}_t^{(2)} = \mathbf{c}_{t-1}^{(2)} \odot \mathbf{f}_t^{(2)} + \mathbf{c}_{t-1}^{(1)} \odot \mathbf{u}_t^{(2)}, \tag{25d}$$
$$\mathbf{o}_t = \boldsymbol{\sigma}\big(\mathbf{W}_o \mathbf{v}_t + \mathbf{b}_o\big), \tag{25e}$$
$$\mathbf{h}_t = \tanh(\mathbf{o}_t \odot \mathbf{c}_t). \tag{25f}$$

RRNN($\mathcal{F}$). Derived from $\mathcal{F}$ (§5.1).

$$\mathbf{f}_t^{(j)} = \boldsymbol{\sigma}\big(\mathbf{W}_f^{(j)} \mathbf{v}_t + \mathbf{b}_f^{(j)}\big), \quad j = 1, 2, \tag{26a}$$
$$\mathbf{u}_t^{(j)} = (1 - \mathbf{f}_t^{(j)}) \odot \mathbf{W}_u^{(j)} \mathbf{v}_t, \quad j = 1, 2, \tag{26b}$$
$$\mathbf{p}^{(j)} = \boldsymbol{\sigma}(\mathbf{b}_p^{(j)}), \quad j = 1, 2, \tag{26c}$$
$$\mathbf{r} = \boldsymbol{\sigma}(\mathbf{b}_r), \tag{26d}$$
$$\mathbf{c}_t^{(1)} = \mathbf{c}_{t-1}^{(1)} \odot \mathbf{f}_t^{(1)} + \mathbf{u}_t^{(1)}, \tag{26e}$$
$$\mathbf{c}_t^{(2)} = \mathbf{c}_{t-1}^{(2)} \odot \mathbf{f}_t^{(2)} + (\mathbf{c}_{t-1}^{(1)} + \mathbf{r}) \odot \mathbf{u}_t^{(2)}, \tag{26f}$$
$$\mathbf{c}_t = \mathbf{p}^{(1)} \odot \mathbf{c}_t^{(1)} + \mathbf{p}^{(2)} \odot \mathbf{c}_t^{(2)} \tag{26g}$$
$$\mathbf{o}_t = \boldsymbol{\sigma}\big(\mathbf{W}_o \mathbf{v}_t + \mathbf{b}_o\big), \tag{26h}$$
$$\mathbf{h}_t = \tanh(\mathbf{o}_t \odot \mathbf{c}_t). \tag{26i}$$

The output gates (Equations 23d, 24d, 25e, and 26h) are optional. They are only used in language modeling experiments, where we empirically find that they improve performance.

E Experimental Setup

E.1 Implementation Details

Our implementation is based on Lei et al. (2017b)[11] and Peng et al. (2018),[12] using PyTorch.[13]

E.2 Language Modeling

For hyperparameters, we do not deviate much from the language modeling experiments in Lei et al. (2017b). We change the hidden sizes for all compared models based on the trainable parameter budget, and adjust the dropout probabilities accordingly to keep the number of remaining hidden units is roughly the same in expectation. Besides, we observe that RRNN($\mathcal{C}$) and RRNN($\mathcal{F}$) fail to converge when optimized with the SGD algorithm using 1.0 initial learning rate. And thus we use 0.5 for both models. Other hyperparameters are kept the same as Lei et al. (2017b).

[11] https://github.com/taolei87/sru
[12] https://github.com/Noahs-ARK/SPIGOT
[13] https://pytorch.org/

Type	**Values**
Hidden size	$[100, 300]$
Vertical dropout	$[0.0, 0.5]$
Recurrent dropout	$[0.0, 0.5]$
Embedding dropout	$[0.0, 0.5]$
Learning Rate	$[10^{-2}, 10^{-4}]$
ℓ_2 regularization	$[10^{-5}, 10^{-7}]$
Gradient Clipping	$[1.0, 5.0]$

Table 6: The hyperparameters explored using random search algorithm in the text classification experiments.

E.3 Text classification

We train our models using Adam (Kingma and Ba, 2015) with a batch size of 16 (for Amazon) or 64 (for the smaller datasets). Initial learning rate and ℓ_2 regularization are hyperparameters. We use 300-dimensional GloVe 840B embeddings (Pennington et al., 2014) normalized to unit length and fixed, replacing unknown words with a special UNK token. Two layer RNNs are used in all cases. For regularization, we use three types of dropout: a recurrent variational dropout, vertical dropout and a dropout on the embedding layer.

We tune the hyperparameters of our model on the development set by running 20 epochs of random search. We then take the best development configuration, and train five models with it using different random seeds. We report the average test results. The hyperparameters values explored are summarized in Table 6. We train all models for 500 epochs, stopping early if development accuracy does not improve for 30 epochs. During training, we halve the learning rate if development accuracy does not improve for 10 epochs.