Supplementary Material For:
Content Selection in Deep Learning Models of Summarization

A Details on Sentence Encoders

We use 200 dimenional word embeddings w_i in all models. Dropout is applied to the embeddings during training. Wherever dropout is applied, the drop probability is .25.

A.1 Details on RNN Encoder

Under the *RNN* encoder, a sentence embedding is defined as $h = [\overrightarrow{h}_{|s|}; \overleftarrow{h}_1]$ where

$$\overrightarrow{h}_0 = \mathbf{0}; \quad \overrightarrow{h}_i = \overrightarrow{\text{GRU}}(w_i, \overrightarrow{h}_{i-1}) \tag{1}$$

$$\overleftarrow{h}_{|s|+1} = \mathbf{0}; \quad \overleftarrow{h}_i = \overleftarrow{\text{GRU}}(w_i, \overleftarrow{h}_{i+1}), \tag{2}$$

and $\overrightarrow{\text{GRU}}$ amd $\overleftarrow{\text{GRU}}$ indicate the forward and backward GRUs respectively, each with separate parameters. We use 300 dimensional hidden layers for each GRU. Dropout is applied to GRU during training.

A.2 Details on CNN Encoder

The *CNN* encoder has hyperparameters associated with the window sizes $K \subset \mathbb{N}$ of the convolutional filters (i.e. the number of words associated with each convolution) and the number of feature maps $M_k \in \mathbb{N}$ associated with each filter (i.e. the output dimension of each convolution). The *CNN* sentence embedding h is computed as follows:

$$a_i^{(m,k)} = b^{(m,k)} + \sum_{j=1}^{k} W_j^{(m,k)} \cdot w_{i+j-1} \tag{3}$$

$$h^{(m,k)} = \max_{i \in 1,\ldots,|s|-k+1} \text{ReLU}\left(a_i^{(m,k)}\right) \tag{4}$$

$$h = \left[h^{(m,k)} | m \in \{1,\ldots,M_k\}, k \in K\right] \tag{5}$$

where $b^{(m,k)} \in \mathcal{R}$ and $W^{(m,k)} \in \mathcal{R}^{k \times n'}$ are learned bias and filter weight parameters respectively, and $\text{ReLU}(x) = \max(0, x)$ is the rectified linear unit activation. We use window sizes $K = \{1, 2, 3, 4, 5, 6\}$ with corresponding feature maps sizes $M_1 = 25, M_2 = 25, M_3 = 50, M_4 = 50, M_5 = 50, M_6 = 50$, giving h a dimensionality of 250. Dropout is applied to the CNN output during training.

B Details on Sentence Extractors

B.1 Details on RNN Extractor

$$\overrightarrow{z}_0 = \mathbf{0}; \quad \overrightarrow{z}_i = \overrightarrow{\text{GRU}}(h_i, \overrightarrow{z}_{i-1}) \tag{6}$$

$$\overleftarrow{z}_{n+1} = \mathbf{0}; \quad \overleftarrow{z}_i = \overleftarrow{\text{GRU}}(h_i, \overleftarrow{z}_{i+1}) \tag{7}$$

$$a_i = \text{ReLU}\left(U \cdot [\overrightarrow{z}_i; \overleftarrow{z}_i] + u\right) \tag{8}$$

$$p(y_i = 1|h) = \sigma\left(V \cdot a_i + v\right) \tag{9}$$

where $\overrightarrow{\text{GRU}}$ and $\overleftarrow{\text{GRU}}$ indicate the forward and backward GRUs respectively, and each have separate learned parameters; U, V and u, v are learned weight and bias parameters. The hidden layer size of the GRU is 300 for each direction and the MLP hidden layer size is 100. Dropout is applied to the GRUs and to a_i.

B.2 Details on Seq2Seq Extrator

$$\overrightarrow{z}_0 = \mathbf{0}; \quad \overrightarrow{z}_i = \overrightarrow{\text{GRU}}_{enc}(h_i, \overrightarrow{z}_{i-1}) \tag{10}$$

$$\overleftarrow{z}_{n+1} = \mathbf{0}; \quad \overleftarrow{z}_i = \overleftarrow{\text{GRU}}_{enc}(h_i, \overleftarrow{z}_{i+1}) \tag{11}$$

$$\overrightarrow{q}_i = \overrightarrow{\text{GRU}}_{dec}(h_i, \overrightarrow{q}_{i-1}) \tag{12}$$

$$\overleftarrow{q}_i = \overleftarrow{\text{GRU}}_{dec}(h_i, \overleftarrow{q}_{i+1}) \tag{13}$$

$$q_i = [\overrightarrow{q}_i; \overleftarrow{q}_i], \;\; z_i = [\overrightarrow{z}_i; \overleftarrow{z}_i] \tag{14}$$

$$\alpha_{i,j} = \frac{\exp\left(q_i \cdot z_j\right)}{\sum_{j=1}^{n} \exp\left(q_i \cdot z_j\right)}, \;\; \bar{z}_i = \sum_{j=1}^{n} \alpha_{i,j} z_j \tag{15}$$

$$a_i = \text{ReLU}\left(U \cdot [\bar{z}_i; q_i] + u\right) \tag{16}$$

$$p(y_i = 1|h) = \sigma\left(V \cdot a_i + v\right). \tag{17}$$

The final outputs of each encoder direction are passed to the first decoder steps; additionally, the first step of the decoder GRUs are learned "begin decoding" vectors $\overrightarrow{q}_0$ and $\overleftarrow{q}_0$ (see Figure 1.b). Each GRU has separate learned parameters; U, V and u, v are learned weight and bias parameters. The hidden layer size of the GRU is 300 for each direction and MLP hidden layer size is 100. Dropout with drop probability .25 is applied to the GRU outputs and to a_i.

B.3 Details on Cheng & Lapata Extractor.

The basic architecture is a unidirectional sequence-to-sequence model defined as follows:

$$z_0 = \mathbf{0}; \quad z_i = \text{GRU}_{enc}(h_i, z_{i-1}) \tag{18}$$

$$q_1 = \text{GRU}_{dec}(h_*, z_n) \tag{19}$$

$$q_i = \text{GRU}_{dec}(p_{i-1} \cdot h_{i-1}, q_{i-1}) \tag{20}$$

$$a_i = \text{ReLU}\left(U \cdot [z_i; q_i] + u\right) \tag{21}$$

$$p_i = p(y_i = 1|y_{<i}, h) = \sigma\left(V \cdot a_i + v\right) \tag{22}$$

where h_* is a learned "begin decoding" sentence embedding (see Figure 1.c). Each GRU has separate learned parameters; U, V and u, v are learned weight and bias parameters. Note in Equation 20 that the decoder side GRU input is the sentence embedding from the previous time step weighted by its probabilitiy of extraction (p_{i-1}) from the previous step, inducing dependence of each output y_i on all previous outputs $y_{<i}$. The hidden layer size of the GRU is 300 and the MLP hidden layer size is 100. Dropout with drop probability .25 is applied to the GRU outputs and to a_i.

Note that in the original paper, the Cheng & Lapata extractor was paired with a *CNN* sentence encoder, but in this work we experiment with a variety of sentence encoders.

B.4 Details on SummaRunner Extractor.

Like the RNN extractor it starts with a bidirectional GRU over the sentence embeddings

$$\overrightarrow{z}_0 = \mathbf{0}; \quad \overrightarrow{z}_i = \overrightarrow{\text{GRU}}(h_i, \overrightarrow{z}_{i-1}) \tag{23}$$

$$\overleftarrow{z}_{n+1} = \mathbf{0}; \quad \overleftarrow{z}_i = \overleftarrow{\text{GRU}}(h_i, \overleftarrow{z}_{i+1}), \tag{24}$$

It then creates a representation of the whole document q by passing the averaged GRU output states through a fully connected layer:

$$q = \tanh\left(b_q + W_q \frac{1}{n} \sum_{i=1}^{n} [\overrightarrow{z}_i; \overleftarrow{z}_i]\right) \tag{25}$$

A concatentation of the GRU outputs at each step are passed through a separate fully connected layer to create a sentence representation z_i, where

$$z_i = \text{ReLU}\left(b_z + W_z[\overrightarrow{z}_i; \overleftarrow{z}_i]\right). \tag{26}$$

The extraction probability is then determined by contributions from five sources:

$$\textit{content} \quad a_i^{(con)} = W^{(con)} z_i, \tag{27}$$

$$\textit{salience} \quad a_i^{(sal)} = z_i^T W^{(sal)} q, \tag{28}$$

$$\textit{novelty} \quad a_i^{(nov)} = -z_i^T W^{(nov)} \tanh(g_i), \tag{29}$$

$$\textit{position} \quad a_i^{(pos)} = W^{(pos)} l_i, \tag{30}$$

$$\textit{quartile} \quad a_i^{(qrt)} = W^{(qrt)} r_i, \tag{31}$$

where l_i and r_i are embeddings associated with the i-th sentence position and the quarter of the document containing sentence i respectively. In Equation 29, g_i is an iterative summary representation computed as the sum of the previous $z_{<i}$ weighted by their extraction probabilities,

$$g_i = \sum_{j=1}^{i-1} p(y_j = 1 | y_{<j}, h) \cdot z_j. \tag{32}$$

Note that the presence of this term induces dependence of each y_i to all $y_{<i}$ similarly to the Cheng & Lapata extractor.

The final extraction probability is the logistic sigmoid of the sum of these terms plus a bias,

$$p(y_i = 1 | y_{<i}, h) = \sigma \left(\begin{array}{l} a_i^{(con)} + a_i^{(sal)} + a_i^{(nov)} \\ + a_i^{(pos)} + a_i^{(qrt)} + b \end{array} \right). \tag{33}$$

The weight matrices W_q, W_z, $W^{(con)}$, $W^{(sal)}$, $W^{(nov)}$, $W^{(pos)}$, $W^{(qrt)}$ and bias terms b_q, b_z, and b are learned parameters; The GRUs have separate learned parameters. The hidden layer size of the GRU is 300 for each direction z_i, q, and g_i have 100 dimensions. The position and quartile embeddings are 16 dimensional each. Dropout with drop probability .25 is applied to the GRU outputs and to z_i.

Note that in the original paper, the SummaRunner extractor was paired with an *RNN* sentence encoder, but in this work we experiment with a variety of sentence encoders.

C Ground Truth Extract Summary Algorithm

Algorithm 1: ORACLEEXTRACTSUMMARYLABELS

Data: input document sentences $s_1, s_2, \ldots, s_n$,
human reference summary R,
summary word budget c.

$y_i := 0 \quad \forall i \in 1, \ldots, n$ // Initialize extract labels to be 0.

$S := [\,]$ // Initialize summary as empty list.

while $\sum_{s \in S} \text{WORDCOUNT}(s) \leq c$ **do** // While summary word count $\leq$ word budget.

/* Add the next best sentence to the summary if it will improve the ROUGE score otherwise no improvement can be made so break. */

$\hat{i} = \arg\max_{\substack{i \in \{1,\ldots,n\}, \\ y_i \neq 1}} \text{ROUGE}(S + [s_i], R)$

if $\text{ROUGE}(S + [s_{\hat{i}}], R) > \text{ROUGE}(S, R)$ **then**

$S := S + [s_{\hat{i}}]$ // Add $s_{\hat{i}}$ to the summary sentence list.

$y_{\hat{i}} := 1$ // Set the $\hat{i}$-th extract label to indicate extraction.

else

break

Result: extract summary labels $y_1, \ldots, y_n$

D Optimizer and initialization settings.

We use a learning rate of .0001 and a dropout rate of .25 for all dropout layers. We also employ gradient clipping ($-5 < \nabla_\theta < 5$). Weight matrix parameters are initialized using Xavier initialization with the normal distribution (Glorot and Bengio, 2010) and bias terms are set to 0. We use a batch size of 32 for all datasets except AMI and PubMed, which are often longer and consume more memory, for which we use sizes two and four respectively. For the Cheng & Lapata model, we train for half of the maximum epochs with teacher forcing, i.e. we set $p_i = 1$ if $y_i = 1$ in the gold data and 0 otherwise when computing the decoder input $p_i \cdot h_i$; we revert to the predicted model probability during the second half training.