談話内における局所文脈の動的分散表現

小林 颯介　　田 然　　岡崎 直観　　乾 健太郎
東北大学 情報科学研究科
{sosuke.k, tianran, okazaki, inui}@ecei.tohoku.ac.jp

1 はじめに

コンピュータの文章読解力を測る一つの方法として、文章の内容に関する質問応答が挙げられる。例えば、次のような文章 (1t) と穴埋め形式の質問文 (1q) を考える。

(1) t. *John is the president of the U.S.*
q. *[X] is the president.*

このような質問応答では、まず質問文 (1q) のプレースホルダ *[X]* に関する局所文脈（この例では *is the president*）を求め、それと意味的に類似した局所文脈を持つ名詞句（解答候補）を探すのが基本的なアプローチである。近年はこの局所文脈の照合に分散表現を用いるアプローチに注目が集まっている [14]。

(2) t. *John is the president of the U.S.*
Jacqueline is the wife of John.
q. *[X] is the wife of the president.*

一方、情報源が文章であるとすると、文章中の複数の情報を組み合わせて初めて解答できる質問も考えられる。例えば、(2) のような質問に答えるには、(2t) の 1 文目の *John* の局所文脈 *is the president* と 2 文目の *Jacqueline* の局所文脈 *is the wife of John* を組み合わせて *is the wife of John, who is the president* のような情報を解答候補 *Jacqueline* の局所文脈として計算する必要がある。

こうした情報の組み合わせ（集約）が上手くモデル化できれば単なる質問応答を越えて談話の理解に一歩近づくと考えられるが、これまでの分散表現に基づく質問応答の研究ではこうした現象を扱えていない。

本稿では、文章中の局所文脈を複数組み合わせて質問に解答することができるモデルの構築を目的として、新しいニューラルネットベースの手法を 2 つ提案する。各解答候補が持つ複数の局所文脈を重み付き平均で集約する方法、および各解答候補の局所文脈の分散表現を談話の進行に従って動的に計算する方法である。近年公開された大規模な質問応答データセット CNN QA[4] で評価実験を行ったところ、提案手法は正答率の向上に寄与し、既報の最高正解率を上回る性能を達成した。

2 データセット

本稿では評価実験のために、Hermann らが近年公開した CNN QA データセット [1][4] を用いる。図 1 に示すように、〈文章, 穴埋め質問文, 答え〉の三つ組を 1 問として

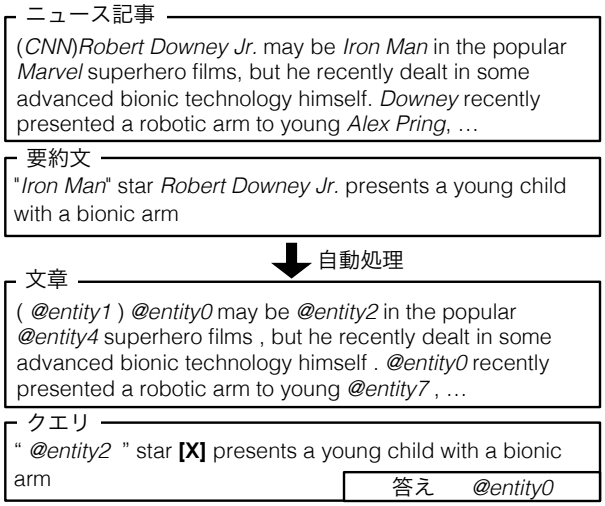

図 1: CNN QA の問題例。下部が実際の QA データ。

おり、問題はすべてニュースサイト [2] の記事文章とその要約文を用いて自動で構築したものである。なお、特徴的な点として全ての固有表現（例. “*Robert Downey Jr.*”, “*Downey*”）は共参照を保ったままランダムな変数表現（例. *@entity0*）へと置換されている。そのため、事前の背景知識の影響を抑えた、より純粋な文章読解力のテストを行えるようになっている。訓練用データには約 38 万問（約 9 万記事）、開発用データ（Valid）及びテスト用データ（Test）には約 3 千問（約 1 千記事）を含む。平均すると、記事内には約 25 種類のエンティティの変数表現を含み、記事の長さは約 700 語である。

3 提案手法

3.1 局所文脈のエンコード

はじめに個々の解答候補の個々の出現（mention）について、その局所文脈を分散表現（ベクトル）にエンコードすることを考える。これには双方向 LSTM[6] を用いる。双方向 LSTM は単語列の情報を分散表現にエンコードするのにしばしば用いられる方法で、次の漸化式 (1)(2) で与えられる。

$$\vec{\boldsymbol{h}}_{c,t} = \overrightarrow{LSTM}(\boldsymbol{x}_{c,t}, \vec{\boldsymbol{h}}_{c,t-1}) \quad \text{(順方向)} \qquad (1)$$
$$\overleftarrow{\boldsymbol{h}}_{c,t} = \overleftarrow{LSTM}(\boldsymbol{x}_{c,t}, \overleftarrow{\boldsymbol{h}}_{c,t+1}) \quad \text{(逆方向)} \qquad (2)$$

[1] https://github.com/deepmind/rc-data

[2] http://www.cnn.com/

紙面の制約から詳細は割愛するが、(1) の $\vec{\boldsymbol{h}}_{c,t}$ は、文 c の 1 番目の単語（文頭記号）から t 番目の単語までの局所文脈をエンコードした分散表現である。同様に (2) の $\overleftarrow{\boldsymbol{h}}_{c,t}$ は、文 c の文末の単語（文末記号）から t 番目の単語までの局所文脈をエンコードした分散表現である。

これらを使って文 c における解答候補 e の出現（τ 番目の単語とする）に対する局所文脈の分散表現（ベクトル）$\boldsymbol{d}_{e,c}$ を図 2 のように計算する。まず、解答候補の出現の左右文脈 $\vec{\boldsymbol{h}}_{c,\tau}$、$\overleftarrow{\boldsymbol{h}}_{c,\tau}$ を計算する。次に、文頭から文末までの単語列のベクトル $\vec{\boldsymbol{h}}_{c,T}$ と文末から文頭までのベクトル $\overleftarrow{\boldsymbol{h}}_{c,1}$ を計算する。最後に、これら 4 つのベクトルを結合し、次式の変換を施して $\boldsymbol{d}_{e,c}$ を得る。

$$\boldsymbol{d}_{e,c} = \tanh(W_{hd}[\vec{\boldsymbol{h}}_{c,\tau}, \overleftarrow{\boldsymbol{h}}_{c,\tau}, \vec{\boldsymbol{h}}_{c,T}, \overleftarrow{\boldsymbol{h}}_{c,1}] + \boldsymbol{b}_d) \quad (3)$$

$\boldsymbol{d}_{e,c}$ は、対象エンティティを囲むような左右の文脈に加えて文全体の情報を捉えたような意味表現になっている。なお、W_{hd} は行列、$\boldsymbol{b}_d$ はバイアスベクトルであり、いずれも学習で調整する [3]。

また、質問文 q についても同様に、プレースホルダの位置を τ とするとき、その局所文脈を次式で計算する。

$$\boldsymbol{u}(q) = \tanh(W_{hu}[\vec{\boldsymbol{h}}_{q,T}, \overleftarrow{\boldsymbol{h}}_{q,1}, \vec{\boldsymbol{h}}_{q,\tau}, \overleftarrow{\boldsymbol{h}}_{q,\tau}] + \boldsymbol{b}_u) \quad (4)$$

質問応答では基本的には、$\boldsymbol{u}(q)$ に最も近い局所文脈を持つ解答候補を探せばよい。

3.2 局所文脈の集約

次に、同じ解答候補 e が談話内で複数回出現する状況を考える。それぞれの出現が局所文脈 $\boldsymbol{d}_{e,c}$ を持つので、それらを重み付き平均で集約する。このとき、直感的には、質問文に近い局所文脈により大きな重みを与えるようにすればよいと考えられる。そこで、そうした重み付き平均の制御を最適化する方法として、近年統計的機械翻訳やキャプション生成などに適用され始めたアテンションメカニズム（attention mechanism）[1, 17] を使う。具体的には、まず、質問文 $\boldsymbol{q}$[4] と個々の出現文脈 $\boldsymbol{d}_{e,c'}$ の関連度 $s'_{e,c'}(q)$ を式 (5) で計算し、式 (6) で正規化する [5]。

$$s'_{e,c'}(q) = \boldsymbol{m}^{\mathrm{T}} \tanh(W_{dm}\boldsymbol{d}_{e,c'} + \boldsymbol{q}) + b_s \quad (5)$$

$$s_{e,c}(q) = \frac{\exp(s'_{e,c}(q))}{\sum_{c'} \exp(s'_{e,c'}(q))} \quad (6)$$

ここで得られる $s_{e,c}(q)$ は、質問文 q が与えられたときに、解答候補 e の出現のうちどの出現に注目すべきかを数値化したものと解釈できる。アテンションメカニズムではこうして計算される注目の大きさを「アテンション（注目度）」と呼んでおり、この注目度を使って重み付き平均を次式のように計算する。

$$\boldsymbol{v}(e; q, D) = W_{dv}(\sum_{c \in D} s_{e,c}(q)\boldsymbol{d}_{e,c}) + \boldsymbol{b}_v \quad (7)$$

こうして得られるベクトル $\boldsymbol{v}(e; q, D)$ は、質問文 q と文章 D が与えられたときの解答候補 e の局所文脈を集約したものと解釈することができる [6]。したがって、質問応答は、質問文のベクトル $\boldsymbol{u}(q)$ に最も近い局所文脈ベクトル $\boldsymbol{v}(e; q, D)$ を持つ解答候補 e を探す問題として定式化できる。すなわち、q および D に対して解答候補 e が答えとなる条件付き確率 $p(e|q, D)$ を次式で推定することができる。

$$p(e|q, D) \propto \exp(\boldsymbol{v}(e; q, D)^{\mathrm{T}}\boldsymbol{u}(q)) \quad (8)$$

以上を提案モデルの基本形（Basic）とする。

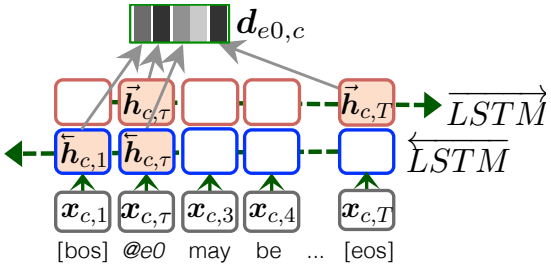

図 2: 文 c における *@e0* の文脈情報 $\boldsymbol{d}_{e0,c}$ のエンコード。

Byway ベクトルによる拡張　上述の基本形モデルでは、各文で局所文脈をまとめ、さらに各解答候補ごとにアテンションメカニズムを適用する。ただし、このままでは、アテンションメカニズムへの誤差伝搬において学習が的確に行われない可能性がある。不正解の解答候補（負例）からアテンションメカニズムに誤差が伝搬するプロセスを考えると、解答候補の推定確率を下げるには、質問文から遠い局所文脈に対する注目度を上げればよいので、何も工夫をしなければ、質問文と似ていない局所文脈に注目を集めるようにアテンションメカニズムが学習されてしまう。アテンションメカニズムは本来質問文と似ている局所文脈に注目を集めるように学習されるべきなので、上のような方向の学習は避けなければならない。この問題は、解答候補のどの出現（mention）にも対応しない空の mention を仮想的に追加し、それに対する仮想的な局所文脈ベクトルを用意することによって解決することができる。ここでは、この仮想的な局所文脈ベクトルを “Byway”（裏道）ベクトルと呼ぶ。“Byway” ベクトルを導入することによって、負例からの誤差伝搬では “Byway” ベクトルに注目が集まるように学習される。しかも、正例の学習の障害にならない。

[3] 添字 hd は、W_{hd} が層 $\boldsymbol{h}$ のベクトルを層 $\boldsymbol{d}$ のベクトルに写像する行列であることを表す。添字 d は、$\boldsymbol{b}_d$ が層 $\boldsymbol{d}$ のベクトルと同じ次元数であることを表す。本稿では以下でもこの表記法を用いる。

[4] ベクトル $\boldsymbol{q}$ は式 (4) のパラメータを変えた同様の計算で求める。

[5] ベクトル $\boldsymbol{m}$、行列 W_{dm}、スカラー値 b_s は、アテンションメカニズムのための学習パラメータである。

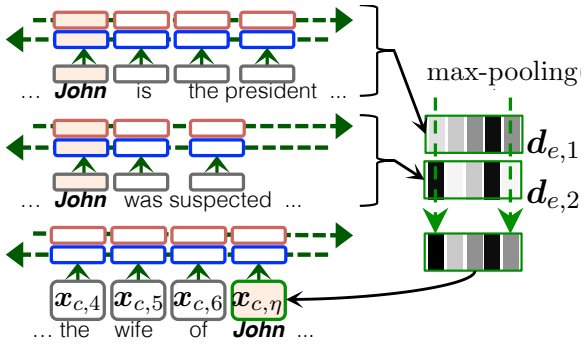

図 3: 複数の文脈情報をマックスプーリングで蓄積し、LSTM への入力 $\boldsymbol{x}_{c,\eta}$ として用いる。

3.3 局所文脈の動的分散表現

3.2 節で導入したアテンションメカニズムで複数の局所文脈を足し合わせられるようになった。しかし、冒頭で述べた (2) の例は局所文脈の足し合わせだけでなく、ときには異なるエンティティ（解答候補）の局所文脈をつなぎ合わせる必要があることを示唆していた。そこで、本稿の 2 つめの提案として、各解答候補の局所文脈を談話の進行に従って動的に計算することによって、局所文脈のつなぎ合わせを実現する方法を考える。

冒頭の例 (2) では、*Jacqueline* の局所文脈 *is the wife of John* に *John* の局所文脈 *is the president* をつなぎ合わせて、*is the wife of John, who is the president* という局所文脈を作ることができれば解答できる。このような局所文脈のつなぎ合わせは、図 3 のように先行文脈の局所文脈ベクトルを現在文の LSTM に入力することによって実現できる。図 3 では、3 文目の局所文脈を計算するときに *John* の入力ベクトルとして、過去 2 回の *John* の局所文脈を用いている。これによって 3 文目の局所文脈 *the wife of John* と *John* の先行局所文脈をつなぎ合わせることができる。

上の例のように先行局所文脈が複数ある場合は、それらをどのように重ね合わせるかがもう一つの問題となるが、ここではマックスプーリング [8] によってベクトルの重ね合わせを行う。マックスプーリングは各要素の時刻や順序の変化に対する頑健性が高いと言われており、我々の目的に適う道具立てだと期待できる。具体的には、解答候補 e が出現したそれまでの文 c' の局所文脈 $\boldsymbol{d}_{e,c'}$ すべてにわたって、各次元について最大値をとる。以上を総合すると、解答候補 e が後続文 c の位置 η に現れる時、それに対応する LSTM への入力を次式で与える。

$$\boldsymbol{x}_{c,\eta} = W_{dx} \max_{c' \prec c}\text{-pooling}(\boldsymbol{d}_{e,c'}) + \boldsymbol{b}_x$$

[6]実際には、式 (7) 内ではバイアスベクトル $\boldsymbol{b}_v$ の他に、「エンティティが質問文内に既に現れている」場合に足し合わせるヒューリスティック用のベクトル $\boldsymbol{b}_{v'}$ も存在するが、ここでは説明の簡略化のために省略した。

モデル	開発	テスト
Basic Proposed Model (Basic)	0.585	0.612
Basic + Max-pooling	0.674	0.683
Basic + Byway	0.664	0.670
Basic + Byway, Max-pooling (Full)	0.675	0.689
Full + w2v-initialization	**0.681**	**0.698**
Deep LSTMs*	0.550	0.570
Attentive Reader*	0.616	0.630
Impatient Reader*	0.618	0.638
Memory Networks**	0.635	0.684
+ Ensemble (11 models)**	0.662	0.694

表 1: CNN QA における正解率。$*$ の結果は Hermann ら [4]、$**$ は Hill ら [5] からの引用である。

4 実験

4.1 実験設定

提案手法の効果を測るため、CNN QA データセットを用いて性能評価実験を行った。配布されている CNN QA の記事文章データは文境界がない単語列となっている。提案モデルでは文ごとに処理を行うため、句読点を用いた単純なヒューリスティックで文を独自に分割し、文頭と文末にシンボル単語を追加した。また、モデルのハイパーパラメータは開発用データで簡単にチューニングした[7]。学習時は推定確率の交差エントロピー誤差を最小化した。なお、我々の提案モデルはすべて `Chainer`[8][16] によって実装した。

4.2 実験結果

表 1 に各モデルの正解率を示す。まず、“Byway” ベクトルの追加によって大きな性能向上が見られる。そして、マックスプーリングによる動的分散表現モデルの場合にもモデルの性能が劇的に向上しており、提案手法の有用性を示している。それら 2 つを組み合わせて使用した場合（Full）にはさらに性能が向上した。さらに、訓練済みの word2vec[9][11] を用いて単語ベクトルを初期化したところ [10]、

表 1 の下段には CNN QA に対する既存の state-of-the-art 手法の性能を掲載した。このうち、Attentive Reader と Impatient Reader[4] は、我々同様に双方向 LSTM とアテンションメカニズムを用いているが、これらのモデルは、全ての解答候補の全ての出現から注目すべき出現を選択するモデルと解釈できる。一方、我々の基本モデルは、解答候補ごとに注目すべき出現を選択するモデルになっ

[7]ベクトルの次元数: 300, Dropout 率: 0.3, バッチサイズ: 50, 最適化手法: RMSProp with momentum [15, 3] (momentum: 0.9, decay: 0.95), 学習率: 1e-4 からスタートさせ、データセット 1 周毎に半減, 勾配クリッピングの上限ノルム: 10. 単語ベクトルは [-0.05, 0.05] の一様分布で初期化し, その他の学習を行う行列は平均 0、分散 2/(列数 + 行数) でのガウス分布で初期化した。

[8]http://chainer.org/

[9]http://code.google.com/p/word2vec/ 上の GoogleNews-vectors-negative300.bin.gz を用いた。

[10]なお、外部の単語ベクトルを用いても固有表現の単語ベクトルは依然使えないため、タスクの枠を外れて背景知識を使ったことにはならない。

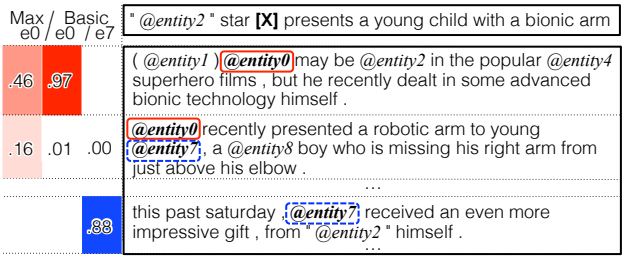

図 4: アテンションメカニズムの挙動の例。各文の注目度重みを文の左に示す。

ている点で異なる。アテンションメカニズムは、アテンションの選択範囲が過度に広いと上手く働かないことが報告されており [10, 17]、この点で我々のモデルの方が有利であると期待できる。また、我々のモデルは解答候補ごとに局所文脈を集約するため、式 (8) のように解答候補の選択を質問文との局所文脈の比較として自然に実現することができる。実際、表 1 に示すように、“Byway” ベクトルを補った我々の基本モデル（Basic+Byway）は、Attentive Reader と Impatient Reader の性能を上回っている。なお、Memory Networks[5] は我々のモデルとは大きく異なっており定性的な比較は難しい。

図 4 に示した問題では正答は*@entity0* であり、1 文目と 2 文目の情報が複合的に含まれたような質問文になっている。Basic モデルでは、この問題に*@entity7* と誤答している。一方で動的分散表現モデルでは 2 文目にも注目度が割り振られ、複合的に情報を用いて正しく*@entity0* と解答できている。

さらに、定量的にも分析を行った。開発用データにおいて、動的分散表現モデルのみで正解した 583 問における正解エンティティの本文中の出現回数の平均値（7.4）は、Basic モデルでも併せて正解した 2064 問における値（6.6）よりも大きかった。これは動的分散表現モデルが、エンティティが多く出現した場合での文脈情報をより適切に統合できるようになったことを示唆している。

5 おわりに

本稿の主たる貢献は、談話内のエンティティの意味表現を動的に生成することの重要性の指摘、及びその具体的手法の提案と検証である。実験の結果、CNN QA において提案手法は性能向上に貢献し、さらには教師無しデータで訓練済みのベクトルで単語ベクトルの初期化を行った場合に世界最高性能となることを示した。単語の表層のみに依存せず各単語の意味表現を変える研究としては Li and Jurafsky [9] や Cheng and Kartsaklis [2] が語義曖昧性解消と multi-sense embeddings を組み合わせる効果を検証しているが、談話全体の文脈を各エンティティについて組織的に考慮した研究は本稿が初である。また、本稿では複数文の情報を捉えるために、蓄積の効果をマックスプーリング [8, 13] で、集約の効果をアテンションメカニズム [1, 14] で実現したが、他のアプローチ [7, 12] も十分考えられる。他のアプローチを用いた場合の効果の検証や CNN QA 以外のタスクへの適用は今後の課題とする。

謝辞

本論文の執筆にあたり貴重なご意見を賜りました、株式会社 Preferred Infrastructure の海野裕也様、株式会社 Preferred Networks の皆様、乾・岡崎研究室の皆様に感謝致します。本研究は、JST CREST の支援を受けたものです。また、本研究は、JSPS 科研費 15H05318 の助成を受けたものです。

参考文献

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*, 2015.

[2] Jianpeng Cheng and Dimitri Kartsaklis. Syntax-aware multi-sense word embeddings for deep compositional models of meaning. In *Proceedings of EMNLP*, pp. 1531–1542, 2015.

[3] Alex Graves. Generating sequences with recurrent neural networks. *CoRR*, Vol. abs/1308.0850, , 2013.

[4] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *NIPS 28*, pp. 1684–1692. 2015.

[5] Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. The goldilocks principle: Reading children’s books with explicit memory representations. *CoRR*, Vol. abs/1511.02301, , 2015.

[6] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, Vol. 9, No. 8, pp. 1735–1780, 1997.

[7] Nal Kalchbrenner and Phil Blunsom. Recurrent convolutional neural networks for discourse compositionality. In *Proceedings of the Workshop on Continuous Vector Space Models and their Compositionality*, pp. 119–126, 2013.

[8] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, Vol. 86, No. 11, pp. 2278–2324, 1998.

[9] Jiwei Li and Dan Jurafsky. Do multi-sense embeddings improve natural language understanding? In *Proceedings of EMNLP 2015*, pp. 1722–1732, 2015.

[10] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of EMNLP 2015*, pp. 1412–1421, 2015.

[11] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *NIPS 26*, pp. 3111–3119, 2013.

[12] Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. Building end-to-end dialogue systems using generative hierarchical neural network models. In *30th AAAI*, 2016. to appear.

[13] Richard Socher, Eric H. Huang, Jeffrey Pennington, Andrew Y. Ng, and Christopher D. Manning. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *NIPS 24*, pp. 801–809. 2011.

[14] Sainbayar Sukhbaatar, arthur szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In *NIPS 28*, pp. 2431–2439. 2015.

[15] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5 - msprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning.*, 2012.

[16] Seiya Tokui, Kenta Oono, Shohei Hido, and Justin Clayton. Chainer: a next-generation open source framework for deep learning. In *Proceedings of Workshop on LearningSys in NIPS 28*, 2015.

[17] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *Proceedings of the 32nd International Conference on Machine Learning*, pp. 2048–2057, 2015.