فهرست

پرسش۱. توصیف تصویر با شبکه ترکیبی..... ۱

۱-۱- مجموعه داده و پیش‌پردازش..... ۱

۱-۱-۱- کاربرد توکن‌های ویژه..... ۱

۱-۱-۲- چرا اعمال padding در یادگیر ی مدل ضروری است؟..... ۲

۱-۲- پیاده‌سازی..... ۲

۱-۲-۱- ابعاد بردار خروجی (feature vector) را بررسی و یادداشت کنید..... ۲

۱-۲-۲- چگونه رمز گذار و رمز گشا را به یک End-to-End تبدیل کنیم که قابلیت آموزش داشته باشد؟..... ۳

۱-۳- آموزش و ارزیابی مدل..... ۳

۱-۳-۱- نمودار خطای داده آموزش و ارزیابی را در طول هر دوره (Epoch) گزارش کنید ... ۴

۱-۳-۲- مقایسه الگوریتم Search Greedy و Search Beam ۵

۱-۳-۳- مقایسه نتایج تولید شده توسط الگوریتم Search Greedy و Search Beam ۶

۱-۳-۴- چند نمونه از خطاهای مدل را شناسایی و تحلیل کنید..... ۸

۱-۴- امتیازی..... ۱۰

شکل‌ها

شکل ۱. نمودار خطا و دقت داده تست و آموزش ۴
شکل ۲. نتایج تست اولیه تولیدشده توسط الگوریتمهای Greedy و Beam..... ۷
شکل ۳. نتایج تولیدشده توسط الگوریتمهای Greedy و Beam ۸

جدول‌ها

جدول ۱. مقایسه الگوریتم Beam Search و Greedy Search ۶
جدول ۲. انواع Bleu ۱۱
جدول ۳. مقادیر BLEU-۱ تا BLEU-۴ روی مجموعه تست ۱۲

پرسش۱. توصیف تصویر با شبکه ترکیبی

تمام کدهای این سوال و نمودارها و نتایج حاصل در فایل ResNet50+LSTM.ipynb در مسیر اصلی فولدر موجود می‌باشد.

۱-۱- مجموعه داده و پیش‌پردازش

۱-۱-۱- کاربرد توکن‌های ویژه

تصویری از فایل دیکشنری به صورت json همراه با توکن‌های ویژه:


```
{
    "<pad>": 0,
    "<sos>": 1,
    "<eos>": 2,
    "<unk>": 3,
    "a": 4,
    "abandon": 5,
    "abandoned": 6,
    "abarrotes": 7,
    "abdomen": 8,
    "ability": 9,
    "aboard": 10, … }
```


<sos> Start of Sentence: شروع جمله، نشان می‌دهد که مدل باید تولید توالی را از این نقطه شروع کند .درtraining ، اولین ورودی به decoder است .در inference نیز اولین توکنی است که به decoder داده می‌شود.

<eos> End of Sentence: پایان جمله، نشان می‌دهد که جمله تمام شده .مدل در حین training یاد می‌گیرد که در پایان جمله این توکن را تولید کند .در مرحله‌ی تولید(inference) ، اگر مدل این توکن را تولید کند، تولید caption متوقف می‌شود.

<pad> Padding Token: پرکننده، برای یکسان‌سازی طول جملات استفاده می‌شود .جملات کوتاه‌تر با این توکن تا طول ثابت (مثلاً ۲۰ کلمه) پر می‌شوند .درtraining ، باید loss برای این توکن‌ها نادیده گرفته شود (با ignore_index=word2idx['<pad>']).

<unk> Unknown Token: کلمه ناشناخته، اگر کلمه‌ای در واژگان (vocabulary) وجود نداشته باشد(Out of Vocabulary) ، با این توکن جایگزین می‌شود .درpractice ، این اتفاق زمانی می‌افتد که واژگان را محدود کرده باشیم یا کلمات نادر در train دیده نشده باشند.

۱-۱-۲- چرا اعمال padding در یادگیر ی مدل ضروری است؟

چون مدل‌های یادگیری عمیق (مخصوصاً LSTM/GRU) با داده‌های batchی کار می‌کنند، و برای اینکه بتوانیم چند نمونه را هم‌زمان پردازش کنیم، طول توالی همه‌ی جملات باید یکسان باشد.

فرض کن معلمی داری که می‌خواهد چند جمله را هم‌زمان بخواند و بررسی کند. اگر طول جمله‌ها متفاوت باشد، خواندن آن‌ها با هم سخت می‌شود. معلم مجبور می‌شود وسط جمله‌ای متوقف شود یا یکی را تا آخر بخواند و منتظر دیگری بماند.

در مدل‌های یادگیری هم همین‌طور است:

اگر جمله‌ای ۳ کلمه باشد و دیگری ۷ کلمه، مدل نمی‌داند چطور هم‌زمان آن‌ها را پردازش کند. پس با اضافه‌کردن کلمه‌های خنثی (padding) در پایان جمله‌های کوتاه، طول همه را برابر می‌کنیم.

این ۲ مثال را در نظر بگیریم:

a dog runs = length: 3

a man with a ball = length: 5

اگر بخواهیم این دو را در یک batch قرار دهیم، باید آن‌ها را به طول مشترک برسانیم، مثلاً طول ۶:

["<sos> a dog runs <eos> <pad> <pad>"]

["<sos> a man with a ball <eos>"]

و در محاسبه‌loss، با ignore_index=word2idx['<pad>'] مشخص می‌کنیم که padding در یادگیری نقش نداشته باشد.

پس Padding ضروری است چون بدون آن مدل نمی‌تواند توالی‌های با طول‌های مختلف را به‌صورت batchآموزش دهد و ما باید طول همه‌ی توالی‌ها را به‌طور مصنوعی برابر کنیم.

۱-۲- پیاده‌سازی

۱-۲-۱- ابعاد بردار خروجی (feature vector) را بررسی و یادداشت کنید

مدل ResNet50 از لایه‌های زیر تشکیل شده:

```
[Conv1 → BN → ReLU → MaxPool]
→ [ResBlock x 16]
→ [AdaptiveAvgPool2d(1x1)]
→ [Flatten]
→ [FullyConnected: 2048 → 1000]  ← classification
```

برای استفاده از ResNet50 به عنوان encoder در مدلcaptioning ، باید:

```
resnet.fc = nn.Linear(2048, 1000)
```

این خط حذف شود و و به‌جای آن فقط لایه‌های feature extractor را نگه داریم و بعد از حذف لایه‌های پایانی خروجی مدل می‌شود:

```
خروجی: tensor با ابعاد (B, 2048, 7, 7)
```

که در آن B اندازه batch است، ۲۰۴۸ تعداد کانال‌های ویژگی است و ۷X۷ ابعاد مکانی خروجی نهایی است. پس ابعاد feature vector برابر با ۲۰۴۸ است.

۱-۲-۲- چگونه رمز گذار و رمز گشا را به یک End-to-End تبدیل کنیم که قابلیت آموزش داشته باشد؟

برای طراحی یک سیستم تولید کپشن به‌صورت سراسری و یکپارچه(End-to-End) ، ضروری است که دو بخش اصلی مدل یعنی رمزگذار (Encoder) و رمزگشا (Decoder) به‌گونه‌ای در کنار یکدیگر قرار گیرند که بتوانند به‌صورت هم‌زمان و هماهنگ آموزش ببینند. در این پروژه، این هدف با تعریف یک کلاس سفارشی تحت عنوان HybridModel محقق شده است. در این ساختار، ابتدا تصویر ورودی توسط رمزگذار که یک شبکه‌ی ResNet50 پیش‌آموزش‌دیده و بدون لایه‌ی Fully Connected است پردازش می‌شود و از آن ویژگی‌های بصری تصویر استخراج می‌گردد. سپس این ویژگی‌ها به عنوان ورودی به رمزگشا داده می‌شود. رمزگشا که ترکیبی از شبکه‌های LSTM و GRU است، به کمک این بردار ویژگی، توالی کلمات توصیف‌کننده‌ی تصویر را مرحله‌به‌مرحله تولید می‌کند. خروجی مدل در هر گام، توزیع احتمالاتی واژگان برای پیش‌بینی کلمه‌ی بعدی است. این طراحی به گونه‌ای انجام شده است که امکان محاسبه‌ی خطا با استفاده از توابع زیان استاندارد مانند CrossEntropyLoss فراهم شده و مدل بتواند با استفاده از روش‌هایی مانند بهینه‌ساز Adam به‌روزرسانی شود. بدین ترتیب، ساختار پیشنهادی به یک مدل End-to-End کامل تبدیل شده که امکان آموزش هم‌زمان رمزگذار و رمزگشا و یادگیری پیوسته از تصویر تا تولید جمله را فراهم می‌سازد. در واقع با تعریف کلاس HybridModel، مدل به یک ساختار End-to-End قابل آموزش تبدیل شده است که امکان یادگیری هم‌زمان رمزگذار و رمزگشا را فراهم می‌سازد.

۱-۳- آموزش و ارزیابی مدل

تنظیمات مقاله:

```
embed_dim = ۲۵۶
```

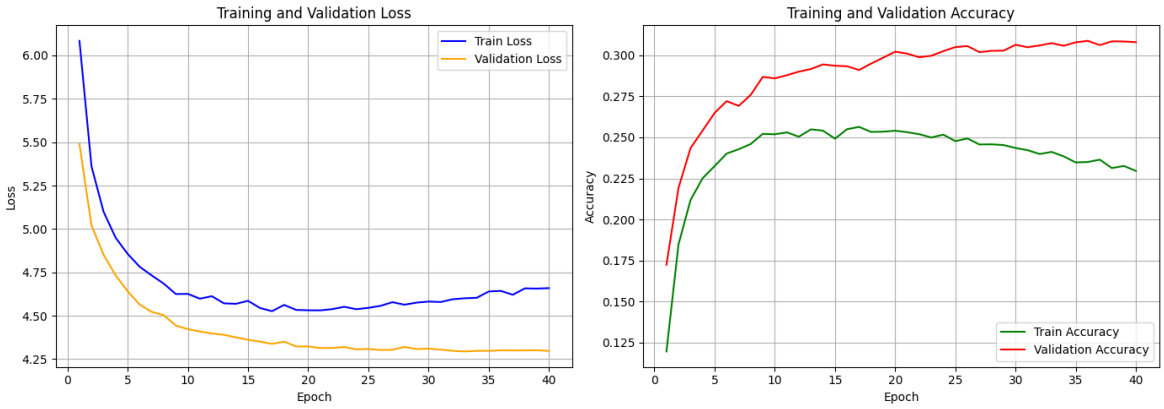
```
hidden_dim = ۵۱۲
vocab_size = len(word2idx)
num_epochs = ۴۰
batch_size = ۳۲
learning_rate = ۰.۰۰۱
```

در مقاله از بهینه‌ساز Adam استفاده شده است و بهترین Beam Width را ۵ بدست آوردند.

ما برای آموزش از learning_rate = 0.0005 استفاده کردیم چون مقدار بزرگ‌تر مثل 1e-3 باعث نوسان loss می‌شد و مقدار کوچک‌تر مثل 1e-4 بسیار کند بود. بنابراین Learning rate برابر ۰.۰۰۰۵ انتخاب شد تا تعادلی بین سرعت همگرایی و پایداری گرادیان‌ها در مدل ترکیبی ResNet50 + LSTM/GRU برقرار شود. هم‌چنین از تابع ReduceLROnPlateau بعنوان scheduler برای مدیریت نرخ یادگیری در طول آموزش.

هم چنین در مدل خود از Teacher Forcing استفاده کرده ایم چون در ادامه، برای اینکه مدل به تولید توکن‌های خودش عادت کند (و بتواند در زمان تست به‌درستی عمل کند)، باید به‌تدریج وابستگی‌اش به داده‌های واقعی کاهش یابد. بنابراین، ابتدا با احتمال ۱ (۱۰۰درصد) از داده واقعی استفاده می‌شود و این احتمال به‌تدریج تا ۰/۵ (۵۰ درصد) کاهش می‌یابد تا مدل به‌صورت طبیعی‌تر یاد بگیرد این تکنیک کمک می‌کند گذار نرم‌تری بین فاز آموزش و فاز تولید واقعی کپشن داشته باشیم.

۱-۳-۱- نمودار خطای داده آموزش و ارزیابی را در طول هر دوره (Epoch) گزارش کنید



شکل ۱. نمودار خطا و دقت داده تست و آموزش

- نمودار Training وValidation Loss

در هر دو نمودار، Loss به‌ویژه تا حدود Epoch ۱۵ به‌سرعت کاهش یافته و سپس به‌تدریج تثبیت شده است. از حدود Epoch ۲۰ به بعد، loss تقریباً نوسانی در بازه باریک (۴.۲۹ تا ۴.۳۵) باقی می‌ماند. مدل به‌خوبی آموزش دیده و loss همگرایی داشته است هم‌چنین Overfitting با توجه به اینکه Validation Loss پایین‌تر از Train Loss باقی مانده، رخ نداده است.

- نمودار Training وValidation Accuracy

دقت (accuracy) مدل به‌سرعت از تقریبا ۰.۱۲ در Epoch۱ به حدود ۰.۳۰ در Val Accuracy رسیده است (تا حدود Epoch۲۰). Train Acc از Epoch۲۰ به بعد، کمی افت کرده ولی Validation Accuracyدر حد بالاتر تثبیت شده. افت جزئی دقت آموزشی نشانه کاهش Teacher Forcing است (مدل به پیش‌بینی مستقل عادت کرده). تثبیت Validation Acc نشانه یادگیری پایدار است.

- ارزیابی:

با بررسی نمودارهای loss و accuracy و همچنین مقادیر BLEU به‌دست‌آمده، مدل به شکل موفقی آموزش دیده و به همگرایی رسیده است. کاهش مداوم loss در مراحل ابتدایی و تثبیت آن در مراحل پایانی نشان‌دهنده‌ی یادگیری مؤثر است. همچنین رشد تدریجی دقت وBLEU-4 ، بیانگر بهبود تدریجی کیفیت کپشن‌های تولیدی توسط مدل است. عدم وجود overfitting نیز با کاهش Teacher Forcing و به‌کارگیری مکانیزم‌های تنظیم نرخ یادگیری (ReduceLROnPlateau) تضمین شده است.

۱-۳-۲- مقایسه الگوریتم Search Greedy وSearch Beam

الگوریتم Greedy Search:

در هر مرحله، تنها بالاترین احتمال (argmax) از خروجی softmax انتخاب می‌شود. مسیری که طی می‌شود، صرفاً مبتنی بر انتخاب لحظه‌ای بهترین گزینه است. از نظر محاسباتی سریع و کم‌هزینه است ولی ممکن است مسیرهای ضعیف اما محلی را انتخاب کند و از توالی‌های بهتر صرف‌نظر کند.

الگوریتم Beam Search:

در هر مرحله، به‌جای یک گزینه، چند مسیر (beam width) نگه داشته می‌شود و بهترین مسیر نهایی از میان آن‌ها انتخاب می‌شود. به مدل اجازه می‌دهد چندین احتمال را بررسی کرده و گزینه‌ی نهایی را با دید کلی‌تری انتخاب کند. خروجی روان‌تر و معنایی‌تر اما نیازمند محاسبات بیشتر و حافظه بالاتر است.

جدول ۱. مقایسه الگوریتم Beam Search و Greedy Search

معیار	Greedy	Beam Search
سرعت	سریع‌تر	کندتر
کیفیت جمله	محدود	بهتر
BLEU Score	پایین‌تر	بالاتر
پوشش مفهومی	کمتر	بیشتر
تکرار نامطلوب	بیشتر	کمتر

در مجموع، Beam Search کیفیت بالاتری نسبت به Greedy Search ارائه می‌دهد، به‌ویژه در زمینه روانی جملات، تطابق بهتر با ground truth ، و کاهش تکرار واژگان. در کاربردهای حساس، استفاده از Beam Search توصیه می‌شود، هرچند در شرایط real-time ممکن است Greedy به دلیل سرعت بالاتر مفیدتر باشد.

۱-۳-۳- مقایسه نتایج تولید شده توسط الگوریتم Greedy Search و Beam Search

در آزمایش‌ها نشان داده شد که Beam Search نسبت به Greedy:

- BLEU-1تا BLEU-4 بالاتری تولید کرده است.
- در بسیاری از تصاویر، کپشن تولیدی توسط Beam Search روان‌تر، دقیق‌تر، و با استفاده از واژگان متنوع‌تر بود.
- میزان تکرار کلمات (مثل"a a a") در Beam کمتر بود و در مواردی اصلا دیده نمی‌شد.

در دو شکل زیر تکرار کلمات "a a a" در تولید کپشن در الگوریتم Greedy مشهود است هم‌چنین مشاهده میشود که مدل نتوانسته کپشن‌های بامعنی و مفهومی تولید کند و جملات در مواردی سطحی هستند:

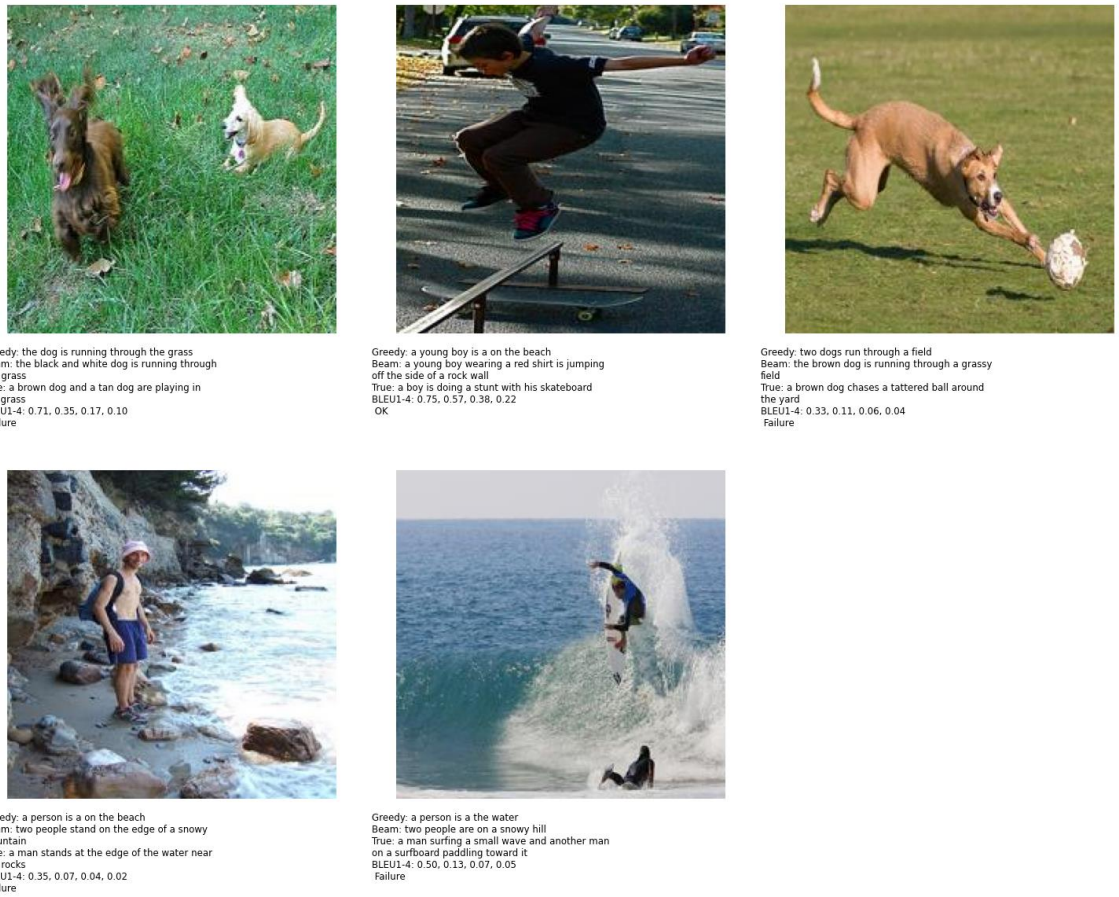
Greedy: a person is a a a a a
Beam: two people are on the beach
True: a man stands at the edge of the water near the rocks
BLEU1-4: 0.38, 0.08, 0.04, 0.03
 Failure

Greedy: two people are a a a a
Beam: the man is wearing a red shirt and blue jeans is standing on front of a brick building
True: a group of children dressed in costume pose for a picture
BLEU1-4: 0.57, 0.14, 0.07, 0.05
 Failure

شکل ۲. نتایج تست اولیه تولیدشده توسط الگوریتمهای Greedy و Beam

برای مقابله با خروجی‌های ضعیف و تکراری (مانند "a a a")، مجموعه‌ای از اقدامات بهینه‌سازی صورت گرفت. با افزودن مکانیزم توجه (Attention) به رمزگشا، مدل توانست در هر گام از تولید جمله، اطلاعات مرتبط‌تری از تصویر استخراج کند. همچنین با اعمال سیاست‌هایی نظیر جلوگیری از تکرار در greedy_search و افزودن جریمه به توالی‌های تکراری در beam_search، خروجی‌ها تنوع و دقت بیشتری پیدا کردند.

به‌علاوه، با بهره‌گیری از تکنیک‌هایی مانند label smoothing ، آموزش مدل به واقعیت تست نزدیک‌تر شد و از وابستگی به واژگان پرتکرار کاسته شد. نتایج BLEU و تحلیل کیفی نیز نشان دادند که این اقدامات به صورت معنادار باعث بهبود روانی و تنوع جملات تولیدشده شدند.



شکل ۳. نتایج تولیدشده توسط الگوریتمهای Greedy و Beam

۱-۳-۴- چند نمونه از خطاهای مدل را شناسایی و تحلیل کنید

با توجه به شکل ۳. نتایج تولیدشده توسط الگوریتمهای Greedy و Beam، مدل Image Captioningدر برخی نمونه‌ها عملکرد مناسبی نداشته و دچار خطاهای مفهومی یا توصیفی شده است:

۱. نمونه ۱ (بالا-چپ)

- تشخیص خطا:

تعداد سگ‌ها به درستی تشخیص داده نشده است (یک سگ گفته شده در حالی که دو سگ در تصویر هستند). رنگ واقعی سگ‌ها (قهوه‌ای و کرم) نادیده گرفته شده و رنگ نادرست (سیاه و سفید) ذکر شده. فعل "playing" که نشان‌دهنده تعامل بین اشیا است، حذف شده.

- تحلیل:

مدل به‌طور محدود اشیای متعدد را توصیف می‌کند، که یکی از نقاط ضعف رایج در مدل‌های captioning کلاسیک است. شاید نبود attention مناسب باعث نادیده‌گرفتن سگ دوم و جزئیات رفتاری شده است. Beam کمی بهتر عمل کرده ولی همچنان مفاهیم کلیدی را حذف کرده است.

۲. نمونه ۲ (بالا-راست)

- تشخیص خطا:

مدل وجود توپ را تشخیص نداده است. فعل مهم "chases" (تعقیب کردن) که نشان‌دهنده حرکت هدفمند است، در جمله نیامده. جزئیات مکانی ("yard") و وصف دقیق از توپ ("tattered") حذف شده‌اند.

- تحلیل:

اشکال در تشخیص اشیای کوچک (مثل توپ) ناشی از ضعف encoder در استخراج ویژگی‌های دقیق یا محدود بودن تمرکزdecoder است. توصیف فعل‌های خاص نیاز به ترکیب اطلاعات زمانی-مکانی دارد که مدل فاقد آن است.

۳. نمونه ۳ (بالا-میانی)

- تشخیص خطا:

Greedy خروجی ناقص و بی‌معنی تولید کرده. Beam به "jumping" اشاره کرده که به مفهوم درست نزدیک‌تر است ولی از "skateboard" که عنصر کلیدی است، غافل مانده.

- تحلیل:

Beam توصیف پویاتری تولید کرده ولی همچنان فاقد واژه‌های خاص و دقیق است. Greedy دچار تکرار و اشتباه دستوری شده که معمولاً از ناتوانی در پیش‌بینی بلندمدت ناشی می‌شود. مشکل در فهم فعالیت خاص (stunt + skateboard) نیاز به یادگیری قوی‌تر از کنش‌ها دارد.

۴. نمونه ۴ (پایین-چپ)

- تشخیص خطا:
  - هر دو خروجی نادرست:
    - Greedy جمله ناقص و غیرطبیعی است.
    - Beam تصویری را توصیف می‌کند که اصلاً وجود ندارد (کوه برفی).
- تحلیل:

ممکن است مدل دچار hallucination (تولید متن از حافظه به جای تصویر) شده باشد. اشتباه Beamمی‌تواند ناشی از شباهت بافت سنگ‌ها به کوه باشد. Greedy نیز نشان‌دهنده نبود کنترل کافی بر ساختار زبانی است.

۵. نمونه ۵ (پایین-راست)

- تشخیص خطا:

فعالیت خاص (surfing) به‌طور کامل حذف شده است. Beam با توصیف "snowy hill" دچار تشخیص نادرست صحنه شده. Greedy هم ساختار دستوری ضعیف دارد و عمل را به‌درستی نشان نمی‌دهد.

- تحلیل:

فعالیت‌هایی مثل "surfing" که نیاز به درک حالت بدن و زمینه دارند، برای مدل چالش‌برانگیز هستند.

Beam احتمالاً از جمله‌ای مشابه آموزش‌دیده‌شده الهام گرفته(hallucination) .

Encoder ممکن است نتواند آب و موج را با دقت از تصویر استخراج کند.

۱-۴- امتیازی

برای ارزیابی کمی عملکرد مدل، از معیار BLEU استفاده شد. این معیار میزان همپوشانی بین کلمات و عبارت‌های کپشن تولیدی و کپشن‌های مرجع انسانی را می‌سنجد. به‌طور خاص، BLEU-1تا-BLEU 4به ترتیب مطابقت‌های unigram تا ۴-gram را بررسی می‌کنند. BLEU-4 سخت‌گیرانه‌ترین معیار است و در این پروژه به عنوان معیار اصلی برای مقایسه روش‌ها (Greedy vs. Beam) استفاده شد.

مدل‌های تولید کپشن برای تصاویر، معمولاً بر اساس مقایسه‌ی خروجی تولیدشده توسط مدل با کپشن‌های مرجع انسانی ارزیابی می‌شوند. معیارهای متعددی برای سنجش کیفیت وجود دارد، از جمله:

1. BLEU (Bilingual Evaluation Understudy)

- معیار اصلی و پرکاربرد
- مبتنی بر مقایسه‌یn-gram های مشترک بین کپشن تولیدشده و کپشن‌های مرجع
- معمولاً از BLEU-1 تا BLEU-4 استفاده می‌شود

2. METEOR

- مشابه BLEU اما دقیق‌تر
- در نظر گرفتن ریشه کلمات، مترادف‌ها، ترتیب کلمات

3. ROUGE (Recall-Oriented)

- بیشتر برای خلاصه‌سازی استفاده می‌شود

- تمرکز بر پوشش کامل واژگان مرجع در خروجی تولیدی

4. CIDEr (Consensus-based Image Description Evaluation)

- معیار تخصصی برای کپشن تصویر
- وزن‌دهی بیشتر به n-gram هایی که در چند مرجع ظاهر شده‌اند
- عملکرد بالاتر نسبت به BLEU در مسائل بصری

BLEUیک معیار عددی بین ۰ تا ۱ (یا ۰ تا ۱۰۰) است که نشان می‌دهد چقدر کلمات و عبارات کپشن تولیدی با کپشن‌های مرجع تطابق دارند.

نحوه محاسبه:

- بر اساس تطابق n-gramها
- میانگین هندسی دقت‌های مختلف (unigram, bigram, …)
- شامل penalty برای جملات خیلی کوتاه (brevity penalty)

جدول ۲. انواع Bleu

BLEU	چه چیزی را می‌سنجد	n-gram
BLEU-1	فقط تطابق واژگان	unigram (تک‌کلمه)
BLEU-2	دقت در ترکیب‌های ساده	bigram (دوکلمه‌ای)
BLEU-3	روانی متوسط و ارتباط جملات	trigram (سه‌کلمه‌ای)
BLEU-4	بیشترین دقت معنایی و ساختاری	Gram-4

در نتایج ما نیز در این تمرین BLEU-1 معمولاً عدد بالاتری دارد (چون فقط تطابق واژگان است)، ولی BLEU-4 دقیق‌تر و سخت‌گیرانه‌تر است و میزان روانی و انسجام جمله را بهتر نشان می‌دهد.

جدول کامل با بررسی و محاسبه BLEU1-4 روی Greedy و Beam و نمایش ۵۰ داده تست در سلول آخر در فایل کدها ResNet50+LSTM.ipynb وجود دارد. آنچه در اینجا آورده شده محاسبه این معیارها برای ۵ تصویر تست گرفته شده است.

جدول ۳. مقادیر BLEU-۱ تا BLEU-۴ روی مجموعه تست

عکس	Bleu-1(G)	Bleu-2(G)	Bleu-3(G)	Bleu-4(G)	Bleu-1(B)	Bleu-2(B)	Bleu-3(B)	Bleu-4(B)
بالا-چپ	۰.۷۱۴۳	۰.۳۴۵۰	۰.۱۶۶۷	۰.۱۰۳۰	۰.۷۴۷۱	۰.۴۵۲۴	۰.۲۷۴۸	۰.۱۵۴۴
بالا-میانی	۰.۷۵۰۰	۰.۵۶۶۹	۰.۳۷۷۰	۰.۲۱۷۳	۰.۶۶۶۷	۰.۴۹۲۴	۰.۴۱۷۴	۰.۳۵۶۶
بالا-راست	۰.۳۳۳۳	۰.۲۵۸۲	۰.۱۴۴۰	۰.۰۹۷۲	۰.۴۲۸۶	۰.۳۷۸۰	۰.۳۰۵۷	۰.۱۹۳۱
پایین-چپ	۰.۳۶۷۹	۰.۲۳۲۷	۰.۰۹۶۳	۰.۰۵۶۰	۰.۳۱۸۵	۰.۰۷۹۲	۰.۰۴۱۳	۰.۰۲۶۱
پایین-راست	۰.۵۰۰۰	۰.۱۳۳۹	۰.۰۷۳۸	۰.۰۴۹۵	۰.۴۹۷۱	۰.۱۱۰۵	۰.۰۵۵۶	۰.۰۳۴۴

با توجه به جدول ۳. مقادیر BLEU-۱ تا BLEU-۴ روی مجموعه تست و جدول موجود در فایل نوت‌بوک در اکثر تصاویر، Beam Search عملکرد بهتری نسبت به Greedy Search در معیارهای BLEU-2تا BLEU-4 داشته است، به ویژه در جملات با ساختار پیچیده‌تر (مثلاً تصویر بالا-راست).

الگوریتمGreedy ، به دلیل انتخاب توکن با بیشترین احتمال در هر مرحله، اغلب جملات ساده، تکراری یا ناقص تولید می‌کند (مثال: "a person is a...").

Beam Search با در نظر گرفتن چندین مسیر ممکن و مقایسه‌ی آن‌ها، موفق شده جملات روان‌تری تولید کند. برای مثال در تصویر بالا-چپ، BLEU-4در Beam حدود ۵۰ درصد بالاتر از Greedy است.

البته در برخی موارد (مثل تصویر بالا-میانی)، Greedy BLEU-1 بالاتر از Beam است که نشان می‌دهد مدل گاهی در انتخاب توکن اول یا دوم بهتر عمل می‌کند، اما در ادامه‌ی جمله Beam عملکرد بهتری دارد.

نتیجه:

Beam Search به طور کلی کیفیت بالاتری در تولید جملات دارد، مخصوصاً برای BLEU-3 و-BLEU 4که نمایانگر ساختار کلی جمله هستند. در صورت نیاز به جملات دقیق‌تر و طبیعی‌تر، Beam با تنظیم مناسب (مثلاً ۳ = width) انتخاب بهتری خواهد بود. با این حال، Greedy سریع‌تر است و در برخی سناریوهای real-time یا حافظه‌ی محدود، کاربردی‌تر خواهد بود.