キーワードによる画像処理システム
Image Processing System with Keywords

情報工学専攻 東 夢太
Yumeta HIGASHI

概要： 本研究では, 画像から特定の物体を消去し, 消去した空間を新たに補間するシステムの開発を行った. 本システムではユーザー入力の軽減および単純化を目的として, 自然言語による画像処理を最終目標と定め, 今回はその足がかりとしてキーワードによる処理領域の指定を行った. また, 画像補間の際, 画像からその説明文および Scene Graph を生成し, それらを利用することで, より違和感のない補間の実現を目指した.
キーワード： 画像処理, 物体検出, キャプション生成, Scene Graph, PatchMatch

1 はじめに

画像加工では多くの場合, その処理領域をユーザーがマウス等で細かに指定する必要がある. そのため, 複数枚の写真に対して同一の処理を行いたいとき, それが手間になる場合がある. 拡大縮小や色変換などの簡単な処理であれば問題にはならないが, 例えば, プライバシー保護で大量の写真から写り込んでいる人物を隠したい場合など, その処理のために一枚一枚ユーザーが細かに領域を指定するのは負担が大きく, 手間になってしまう. 近年, 機械学習技術の進歩によって, 物体検出技術も大きく進歩し, 高速かつ高精度の検出が可能になってきている. そこで, それらの物体検出技術を応用し, 複数画像中の同一カテゴリの物体に対して, 同一の処理を一括して行うシステムを開発することで, ユーザー負担の軽減が可能になると考えられる. 本研究では, ユーザー入力の軽減および単純化を目的として, 自然言語による画像処理を最終目標と定め, 今回はその足がかりとしてキーワードによって処理領域の指定を行うシステムを開発し, その性能評価を行った.

2 提案システム

2.1 システム概要

提案システムは, 画像とキーワードを入力として受け取り, キーワードによって指定された領域に何らかの処理を施したものを最終結果として出力する. 領域指定後の処理としては, 領域内の物体の削除や移動, 入れ替えなど, 様々なタスクが可能であるが, 今回はその処理の一例として, 指定領域を削除しその部分の補間を行う. 以下に処理の流れを示す.

(a) 画像から物体のカテゴリおよびその位置を検出 (YOLOv3 [3])

(b) キーワードによる処理領域の指定

(c) 指定領域と他の物体領域との干渉判定

(d) 干渉領域を含む画像領域の切り出し

(e) 処理 (d) の結果の画像からキャプションを生成 (OBJ2TEXT [4])

(f) キャプションから Scene Graph [2] を生成

(g) Scene Graph から得た位置関係を利用し処理領域を決定

(h) 処理領域内を削除し補間

また, 今後, 物体検出やキャプション生成などの新たな手法が提案された際に, ユーザーがそれらを手軽に本システム上で利用可能とするために, 本システムは各処理に用いる部品を容易に差し替えられるような設計を目指した. 加えて, 各処理の結果を中間結果として保存しておくことも容易であるため, それらを異なるタスクに応用することも可能である.

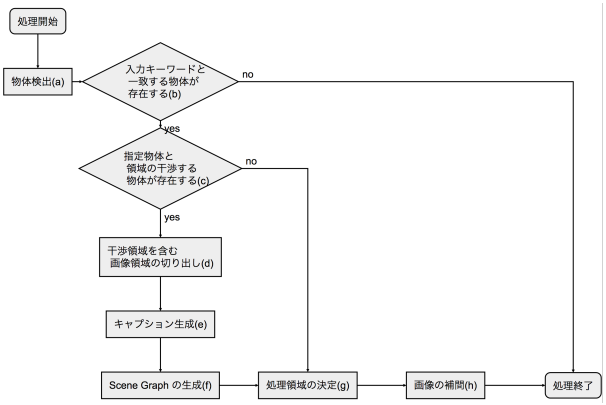

図 1 システム概要

2.1.1 物体検出 (a)

まず, キーワードによる処理領域の指定を行うために, 入力画像から物体のカテゴリおよびその位置 (物体領域) の検出を行う. 提案システムではその手法として, YOLOv3 [3] を採用した. また, 本システムにおいて, ネットワークのパラメータは MS COCO 学習済みモデルを利用した.

2.1.2 キーワードによる処理領域の指定 (b 〜 d)

次に, 入力されたキーワードから処理領域の決定を行う. まず, 物体カテゴリが入力キーワードと一致する物体領域を初期領域として設定する. 初期領域が他の物体領域と干渉していない場合は, そのまま処理領域として決定され, 以降の処理 (d〜g) は行わない. 干渉する物体領域が存在する場合は, 初期領域および干渉領域全てを含む画像領域を切りだし, その画像に対して以降の処理 (e〜g) を行い, 処理領域の更新を行う.

2.1.3 画像キャプション生成 (e)

キーワードで指定された処理領域が他の物体領域と干渉していた場合, 物体同士の関係性を取得するため, まず画像からキャプションを生成する. ここで, 本研究における「画像キャプション」とは,「その画像が表す場面や状況を文章化したもの」とする. キャプション生成には, OBJ2TEXT [4] を用いた. こちらも, パラメータには MS COCO 学習済みモデルを利用した.

2.1.4 Scene Graph の生成 (f)

物体の関係性を解析するため, 2.1.3 節 で得られたキャプションから Scene Graph[2] の生成を行う. Scene Graph 生成の道具として, Stanford Scene Graph Parser を利用した. 図 2 は, “A brown fox chases a white rabbit .” という文章の Scene Graph である. 各ノードの色は赤が object, 緑が attribute, 青が relationship に対応している.

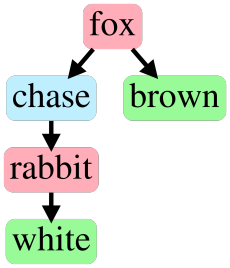

図 2 Scene Graph 例

2.1.5 処理領域の選択 (g)

2.1.4 節 で生成した Scene Graph の relationship を利用し, 物体同士の関係性を考慮する. 本システムでは, グラフで上位に存在する物体ほど画像に対する影響度が高いと考え, その物体がより前景に存在すると判別する. 2.1.1 節 で検出されたカテゴリと Scene Graph の単語が完全には一致しない場合は, それらの類似度によってそれらの紐付けを行う. このとき, 類似度の指標には Path Similarity を用い, その計算の際にシソーラスは WordNet を使用した.

Path Similarity: Path Similarity は, シソーラスの階層構造を利用し, あるワード w_1 のもつ意味 s_1 と, あるワード w_2 のもつ意味 s_2 間の最短パスを求めることで計算できる. 一般的に, より最短パスが短いほど s_1 と s_2 は類似度が高いと考えられている.

2.2 画像の補間 (h)

本研究の主とするところは処理領域の指定までではあるが, その後の画像処理の一例として, 処理領域部分を削除し, 空いた空間を自然に補間するような処理の実装も行った. PatchMatch [1] を用いた類似パッチ (小領域) の探索により, これを実現している. 具体的には, 欠損領域を含む各パッチに対してそれぞれ類似パッチを同一画像中から探索し, 欠損領域中の各ピクセルに対してそれらの色情報の平均を適応して補間を行う.

3 物体検出: YOLO

YOLO (You Only Look Once) とは物体検出アルゴリズムの一種で, その構造はひとつの畳み込みニューラルネットワーク (CNN) で完結しており, 画像から物体領域の推定と, その物体のクラス分類を同時に行えることが大きな特徴となっている. その結果として得られる物体領域は, バウンディングボックス (矩形領域) で表現される.

3.1 アルゴリズム

画像を $S \times S$ のグリッドに分割する. 次に, 分割した各グリッドセルについてそれぞれ B 個のバウンディングボックスとその信頼度を計算する. 各バウンディングボックスは x, y, w, h, 信頼度の 5 つのパラメータからなる. ここで, (x, y) はバウンディングボックスが属するグリッドセルの中心座標, w と h はそれぞれバウンディングボックスの幅と高さを表す. 信頼度には, 物体が存在する確率が反映される. また, 各グリッドセルは, 条件付き確率 $P_r(\text{Class}|\text{Object})$ をもつ. ここで, Object はそのグリッドセルに物体が存在する確率, Class はそのクラスが何であるかを表す. この確率が最大値をとるクラスが, そのグリッドセルのクラスとして設定される. 最後に, 各バウンディングボックスの信頼度と Non-Maximum Suppression の利用によって出力するバウンディングボックスの選択を行う.

4 キャプション生成: OBJ2TEXT [4]

OBJ2TEXT は, 物体レイアウト (カテゴリと位置のペア) から文章を生成する sequence to sequence モデルである. 一般的な Encoder-Decoder 対話モデルでは発話文を Encoder でベクトルに変換し, それを入力として Decoder で応答文を生成する. この OBJ2TEXT では発話文のかわりに, 物体レイアウトから抽出した特徴ベクトルを入力として文章を生成する. 文章の生成には, リカレントニューラルネットワークの一種である LSTM が用いられる. OBJ2TEXT は, 物体レイアウトを Encoder の入力として受け取り, Decoder によってキャプションの生成を行う. 学習では, N 個の物体カテゴリと位置のペア $\{\langle \boldsymbol{o}^{(n)}, \boldsymbol{l}^{(n)} \rangle\}$ およびそれに対応するキャプション $\{\boldsymbol{s}^{(n)}\}$ を与え, 確率的勾配降下方 (SGD) を用いて損失関数の最小化を行うことによって Encoder と Decoder の学習を同時に行う.

4.1 OBJ2TEXT-YOLO

OBJ2TEXT-YOLO は, 画像を入力として受け取り, YOLO で物体検出を行い, 得られた物体レイアウトを利用して OBJ2TEXT でキャプションを生成する.

4.2 OBJ2TEXT-YOLO + CNN-RNN

OBJ2TEXT-YOLO + CNN-RNN モデルでは, 画像を入力として受け取り, 画像の特徴抽出と物体検出の両方を行う. 抽出した特徴ベクトルと検出した物体

レイアウトを同一サイズのベクトルに変換し, それらの合計を用いてキャプション生成を行う.

5 画像の補間 (h)

画像の補間は, 欠損領域がある画像や, 消去したい領域がある画像に対して行われる. ある画像 I に対して, 領域 $R \subset I$ を消去して補間する場合, 消去した領域 R 内の各ピクセルを, 同一画像中の領域 $\overline{R} \subset I$ 内の各ピクセル情報を用いて, “自然に” 補間することが目的となる. ここで言う “自然” とは, 補間された領域に違和感がない, 背景と馴染んでいる状態を表すものとする. 自然な補間を実現するためには, 消去領域の周囲のピクセルと色情報が類似したピクセルを用いて消去領域を埋める手法などがあり, 今回はそれを採用した. そのため, 画像中のあるパッチ (小領域) に対して, ピクセル情報の類似した別のパッチを探索する必要がある. 本研究では, 類似パッチの探索手法には, PatchMatch [1] を使用した.

5.1 PatchMatch

ある画像内の注目パッチに対して, 類似度の高いパッチを類似パッチと呼ぶ. 類似度評価には, 画素値の差分距離による Sum of Squared Difference (SSD) という指標が一般的に用いられる. SSD の値は, 次の式で計算できる.

$$SSD(d_x, d_y) = \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} (I(d_x + x, d_y + y) - T(x, y))^2$$

ここで, $I(x, y)$ は元画像 I の座標 (x, y) における画素値, $T(x, y)$ は注目パッチ画像 T の座標 (x, y) における画素値, w はパッチ画像の幅, h はパッチ画像の高さ, そして d_x, d_y は走査位置, 一般的には左上座標, をそれぞれ表し, $SSD(d_x, d_y)$ の値が小さいほどその類似度は高くなる. 類似パッチ探索の際, 画像 I 中の全ての座標 (d_x, d_y) に対して類似度を計算するのは多くの計算コストがかかってしまう. そこで提案された手法の一つが PatchMatch である. PatchMatch は, 2 枚の画像から類似パッチを探索するためのランダム近似最近傍探索アルゴリズムである. 全探索と比べ, 大幅に探索コストを削減することができる.

5.1.1 アルゴリズム

PatchMatch では, 座標から座標への写像 ($f : \mathbb{R}^2 \to \mathbb{R}^2$) である nearest-neighbor-field (NNF) を定義し, NNF を更新することで類似パッチの探索を高速に行う. ただし, パッチサイズを $w \times h$ としたとき, ここで扱う座標 (x, y) は, (x, y) を左上座標とした $w \times h$ の矩形領域を表すものとする. PatchMatch アルゴリズムは, Initialization, Propagation, Random Search の 3 つのステップで構成される.

Initialization: NNF をランダムに初期化する.

Propagation: 画像の連続性を考慮すると, 画像上で隣接する部位の対応は似た傾向になりやすい. その傾向を利用して, (x, y) の隣接ピクセルの写像先を利用して $f(x, y)$ を更新する.

Random Search: 局所解に陥るのを防ぎ, 対応の精度を上げるため, 現在の $f(x, y)$ を基準に所定の範囲内で再度ランダムに写像先を選択し, そちらの方が類似度が高ければ $f(x, y)$ を更新する.

Initialization の後, Propagation と Random Search を 1 セットとして通常 4〜5 回繰り返す.

5.2 補間手法

本研究の実装では, 消去領域を 1 ピクセルでも含む各パッチに対して, 同一画像の消去領域を含まない各パッチの中から類似パッチを前述の PatchMatch を用いて探索を行う (図 3 (a)〜(c)). その後, 消去領域内の各ピクセルに対して, そのピクセルを含む各パッチの色情報の平均値を求め, その値でそのピクセルを補間する (図 3 (d), (e)).

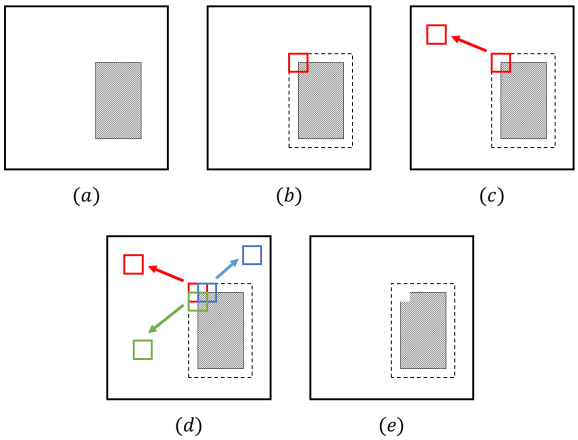

図 3 補間手法

なお, 本システムの実装では, 画像の補間はマスク画像を用いて行うため, 補間領域が矩形領域である必要はない. そのため, より細かな領域指定が可能になれば, 補間の精度も向上すると考えられる.

6 実験

6.1 実験概要

本システムを用いて, 領域が干渉する物体が存在する物体に対してキーワードで領域の指定を行い, 指定された処理領域が妥当であるか評価する. 指定した領域と重なる物体が存在しない場合は, 単純にその領域がそのまま処理領域となるため, その結果はここでは省略する. また, 画像の補間結果は本研究の主題から逸れるため, それに対しては評価は行わない.

6.2 実験結果 1

よく似た二つの実験結果の比較についてここでは述べる. 図 4, 図 5 の 3 枚の画像はそれぞれ上から元画像, キーワードで指定された処理領域を削除した画像, 欠損

領域を補間した画像である．図 4 では “a man riding a horse in field .” というキャプションが生成され，“man ride horse” という関係から “person” が “horse” より前景であると判別している．図 5 では “a man holding a dog on a leash .” というキャプションが生成され，“man hold dog” という関係から “person” が “dog” より前景であると判別しているが，本来は “dog” が前景であると判別するのが適切であると考えられる．

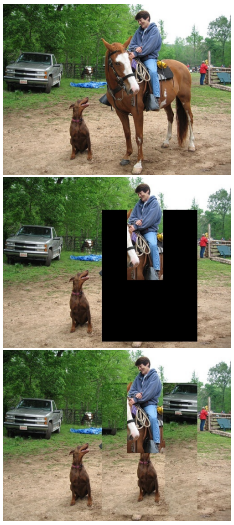

図 4　“horse” で指定

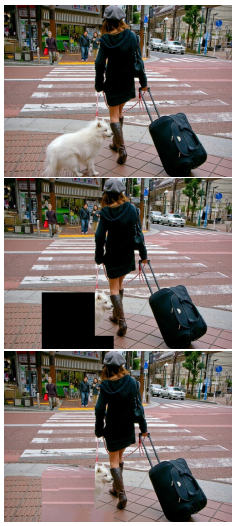

図 5　“dog” で指定

6.2.1　考察 1

この 2 つの結果を比較するだけでも，グラフで上位の物体が前景というのは必ずしも正しいとは限らないことがわかる．各 relationship に対して優先度や前後関係を個別に設定することで，この問題は解決可能であると考えられる．また，生成したキャプション中に指定した物体が現れない場合や，別の物体として扱われてしまう場合も多く見られた．これには，物体の一部が隠れて十分な情報が得られなかった可能性や，頻出する関係性，あまり見られない関係性，など，学習データに偏りがあった可能性が考えられる．

6.3　実験結果 2

実験結果の中でも，特に問題が顕著に現れた例をここに示す．以下の図 6 の画像では，処理対象を “dog” として指定した際，処理領域の更新により，処理領域が完全に消滅してしまった．この画像からは “a woman sitting on a car with a dog on a leash .” というキャプションが生成され，図 8 のような Scene Graph が生成された．結果，“with dog” が “car” にかかっていると判断され，“dog” の領域を完全に包含している “car” の領域が前景であるとして，処理領域の更新を行ってしまっている．

6.3.1　考察 2

対象物体の領域を完全に包含している物体が存在する際に，このような状況が発生しやすいと予想される．この例のように生成キャプションの解釈を誤ってしまった際，本システムではその誤りを検出することができない．結果，誤った関係性から処理領域の更新を行ってしまっている．

図 6　元画像

図 7　検出結果

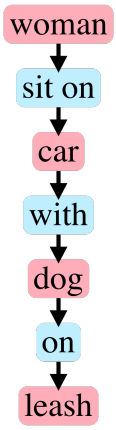

図 8　生成 Scene Graph

7　おわりに

本研究では，画像処理における処理領域の指定を，物体検出の利用によってユーザー入力が単語レベルにまで簡略化されたシステムを開発した．また，画像からキャプションを生成し，それを解析することで物体の位置関係を取得し，処理領域に反映することでより妥当な領域指定の実現を目指した．

実験の結果，期待ほど位置関係を有効に利用することは叶わず，多くの改善点が見つかった．しかし，複雑な処理を複数画像に対して一括して行う枠組みは構築することができた．

参考文献

[1] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, *PatchMatch: a randomized correspondence algorithm for structural image editing*, ACM Trans. Graph., 28 (2009), pp. 24:1–24:11.

[2] J. Johnson, R. Krishna, M. Stark, L. Li, D. A. Shamma, M. S. Bernstein, and F. Li, *Image retrieval using scene graphs*, in IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015, 2015, pp. 3668–3678.

[3] J. Redmon and A. Farhadi, *YOLOv3: An incremental improvement*, CoRR, abs/1804.02767 (2018).

[4] X. Yin and V. Ordonez, *OBJ2TEXT: Generating visually descriptive language from object layouts*, CoRR, abs/1707.07102 (2017).