P23~

すでに3.1.2節で紹介したKarapetyan et al.(2017)とMascia et al.(2014)で使用されたヒューリスティック構築の枠組みは、この目で理解することができる。前述のように、MLが提供するCOアルゴリズムのパラメトリゼーションは非常に広い意味で理解されるべきである。例えば、癌治療のための放射線治療の場合、Mahmoodら(2018)はMLを用いて候補となる治療法を生成し、その後COアルゴリズムによって精緻化され、実現可能な計画となる。すなわち、生成的敵対ネットワーク(GAN)が、CTスキャン画像を潜在的な放射線計画に色付けするために使用され、その後、逆最適化(Ahuja and Orlin, 2001)が、計画を実現可能にするために結果に適用される(Chan et al., 2014)。一般に、GANは2つの異なるネットワークで構成される。1つ（生成器）は画像を生成し、もう1つ（識別器）は生成された画像と実画像のデータセットを識別する。両者は交互に学習される：識別器は通常の教師あり目的を通して学習され、一方、生成器は識別器を欺くように学習される。Mahmoodら(2018)では、特定のタイプのGAN(条件付きGAN)が、ランダム画像の代わりにカラーリングを提供するために用いられている。興味のある読者は、GANの概要についてCreswellら(2018)を参照されたい。我々は、何らかの表現を学習するために用いられるMLモデルが、Kruberら(2017)で用いられる分解統計量や、Bonamiら(2018)におけるLP情報のような、別のCOアルゴリズムによって与えられる情報の断片を特徴として用いることがあることを指摘して、このセクションを終える。さらに、我々は、充足可能性の文脈では、インスタンスの特定のクラスタに対して実行するアルゴリズムのタイプの学習が、アルゴリズム自体のパラメータの学習と対になっていることに注目する。

P24

*3.2.3 Machine learning alongside optimization algorithms*

前節の文脈を最大限に一般化するために、図9に示すように、実行中MLモデルを繰り返し呼び出すCOアルゴリズムを構築することができる。マスターアルゴリズムは、高レベルの構造を制御する一方で、低レベルの決定を支援するためにMLモデルを頻繁に呼び出す。このアプローチと前節で議論した例との重要な違いは、同じMLモデルをCOアルゴリズムが、アルゴリズムの反復回数のオーダーで、同じタイプの決定を何回も行うために使用することである。前節と同様、このようなアルゴリズムの前後に追加のステップを適用することを妨げるものはない。

図９

これは明らかにMILPの分枝限定木の文脈であり、分枝変数を選択するタスクが、発見的すぎるか遅すぎるかのどちらかであり、したがって学習の良い候補であることは既に述べた(Lodi and Zarpellon, 2017)。この場合、一般的なアルゴリズムは、同じソフトウェアアーキテクチャーと下界と上界に関する同じ保証を持つ、分岐と束縛のフレームワークであることに変わりはないが、各ノードで行われる分岐の決定は学習されることになる。同様に、Hottungら(2017)の、ヒューリスティック木探索のための分岐方針と値ネットワークの両方を学習する研究は、紛れもなくこの文脈に適合する。MILPsを解く際のもう一つの重要な側面は、原始的なヒューリスティック、すなわち、成功を保証することなく、実現可能な解を見つけるためにB&Bノードに適用されるアルゴリズムの使用である。その明らかな利点に加え、良い解は解の値により厳しい上限を与え（最小化問題の場合）、より多くの木の刈り込みを可能にする。ヒューリスティックは分岐ノードに依存するため（分岐はいくつかの変数を特定の値に固定するため）、頻繁に実行する必要があります。しかし、ヒューリスティックを頻繁に実行しすぎると、木の探索が遅くなる可能性があります。Khalilら(2017b)は、与えられたヒューリスティックを実行することで、これまでに発見された最良の解よりも良い解が得られるかどうかを予測するMLモデルを構築し、モデルの結果が正であればいつでもそのヒューリスティックを貪欲に実行する。

P25

セクション3.2.1で既に議論したBaltean-Lugojanら(2018)が用いた近似は，最適化問題の解の高レベルの記述，すなわち目的値を予測する例である．それにもかかわらず，目標は元のQPを解くことである．従って、学習されたモデルは、有望な切断面を選択するために繰り返し質問される。MLモデルは有望な切断面を選択するためだけに使用されるが、一度選択された切断面はLPrelaxationに追加されるため、MLの結果は厳密なアルゴリズムに埋め込まれる。このアプローチは、このタイプのアルゴリズムの有望な方向性を強調する。最適な切断面を追加することは、問題を高速に解くために必要であるため、学習された決定は重要である（あるいは、NPhard問題が存在する場合、最適化は意味のある解を得る前にタイムアウトする可能性があるため、十分に高速である）。同時に、近似的な決定（多くの場合、確率の形で）は、アルゴリズムの厳密性を損なわない：追加されたどの切断面も有効であることが保証される。この設定は、MLアルゴリズムによる保証の必要性（活発で困難な研究分野である）を減らしつつ、MLが繁栄する余地を残す。加えて、Larsenら(2018)のアプローチは、MLがリアルタイムで予約決定を行うために反復的に呼び出されるマスターアルゴリズムの一部であることに留意されたい。3.1.2節で紹介したKhalilら(2017a)の研究も、結果として得られるアルゴリズムが発見的であるとしても、この設定に属する。実際、MLモデルは最も関連性の高いノードを選択するように求められ、マスターアルゴリズムは部分ツアーを維持し、その長さなどを計算する。マスターアルゴリズムは非常に単純であるため、3.2.1節で述べられているように、エンドツーエンドの手法としてこの貢献を見ることも可能であるが、ここで行われているように、より一般的に解釈することも可能である。

セクション3.1.2で提示され、前のセクションで言及された、Karapetyan et al.(2017)のヒューリスティックを構築するためのマルコフ連鎖のフレームワークも、繰り返される決定としてフレーミングすることができる。遷移行列は、ある状態から別の状態への遷移、すなわち次の手を選択するという低レベルの決定を行うために、問い合わせとサンプリングが可能である。このセクション3.2で行った3つの区別は、一般的なものであるため、重複する可能性がある。本節を終える前に、リカレントアルゴリズムの決定を学習することは、深層学習の分野でも、例えば、確率的勾配降下において勾配更新を適用する方法を決定するメタ学習の分野で使用されていることに言及する価値がある（Andrychowicz et al, 2016; Li and Malik, 2017;Wichrowska et al.）

P26

4 Learning objects

前節では、COのためのMLの主な貢献を直交的に分類し、既存の文献を概観した。本節では、学習プロセスを駆動する目的を定式化し研究する。

4.1 Multi-instance formulation

以下では、（Bischl et al. (2016)から着想を得た）抽象的な学習定式化を紹介する。MLの実践者はどのように最適化アルゴリズムを比較するのだろうか？Iを問題インスタンスの集合と定義し、PをIに対する確率分布と定義する。実際には、IやPにアクセスすることはできないが、モントリオールの宅配会社で動機づけられたように、Pからいくつかのサンプルを観察することができる。アルゴリズムAの集合に対して、m : I × A ！Rを問題インスタンスに対するアルゴリズムの性能の尺度とする（低いほど良い）。これは、発見された最良の解の目的値とすることもできるが、最適性の境界、結果の不在、実行時間、リソースの使用量などの要素を組み込むこともできる。a1, a2 2 Aを比較する場合、MLの専門家はEi∼P m(i, a1)とEi∼P m(i, a2)を比較する。

（４）式

これらの量を測定することは困難であるため、一般的には、P(i, a1)とP(i, a2)からサンプリングされた独立なインスタンスからなる有限データセットDtrainを用いて、経験的推定を行います。

（５）式

これは直感的であり、実際にも行われていることである。もちろん、このような期待値は、異なるデータセット（異なるI'sとP's）、異なる尺度（異なるm's）に対して計算することができる。

これはすでに学習問題である。我々がリーン化によって解決したいより一般的な問題は

（６）式

P27

２つのアルゴリズム間を比較する代わりに、我々は、数え切れない、多分ノンパラメトリックなアルゴリズム空間間を比較することができる。例えば、B&Bの分岐政策πを学習する場合を考える。この方針を、重みの集合θ 2 Rpを持つニューラルネットワークと定義すると、パラメトリックB&Bアルゴリズムa(π )が得られ、(6)は次のようになる。

（７）式

残念ながら、この問題を解くのは難しい。一方では、パフォーマンス尺度m は微分可能でなく、閉形式がないことがほとんどである。これについてはセクション4.2で説明する。一方、(6)の期待値の計算は困難である。(5)のように、有限のデータセットを用いた経験分布を用いることもできるが、それはセクション4.3で説明するように一般化の考察につながる。

次に進む前に、(6)をより一般的にするために新しい要素を導入しよう。この式は、一度インスタンスが与えられれば、パフォーマンス測定の結果は決定論的であることを示唆している。これは複数の理由から非現実的である。例えば、ハードウェアやシステムに依存する実行時間などである。また、アルゴリズムが確率的であるように設計されている場合、あるいは一部の演算が非決定論的である場合、あるいはアルゴリズムがいくつかの外部パラメータの選択に対してロバストであるべきであるという事実を表現するために、無視できないランダム性の原因を組み込むこともできる。τをランダム性の発生源、π 2を学習される内部方針、a(π, τ ) を結果のアルゴリズムとすると、(6) を次のように再定式化できる。

（８）式

特に、セクション3.2.3のように繰り返し決定を学習する場合、このランダム性の源は、環境p(s′, r|a, s)のダイナミクスを用いて、MDPでたどられる軌跡に沿って表現することができる（図2参照）。(8)の追加は、セクション4.3の一般化に関する議論に役立つ。

P２８

4.2 Surrogate objectives

前節で、我々は適切な学習目的を定式化した。ここでは、その目的をセクション3.1の学習方法、すなわち、実証と経験に関連付けようとする。MLモデルの通常の学習指標、例えば教師あり（模倣）学習における分類の精度が向上しているとすれば、(6)の性能指標も向上しているということだろうか？

セクション3.1.2で調査したように、(8)を解くための簡単なアプローチは、強化学習（直接最適化法を含む）のものである。例えば、(6)のパフォーマンス目標が、(7)のパフォーマンス目標に一致する場合、(8)のように直接最適化することができる。例えば、B&B変数選択ポリシーのパフォーマンス目的が、オープンしたノード数を最小化することである場合、そのポリシーは、ノード数の増加を抑制する報酬を受け取ることができ、したがって、枝刈りにつながる変数を選択するインセンティブを与えることができる。しかし，それは常に可能であるとは限らず，1つの報酬を軌跡の最後まで遅らせるという選択肢しか残らない．このような疎な報酬設定はRLアルゴリズムにとって困難であり、中間的な達成を奨励するために代理の報酬信号を設計したくなるかもしれない。これは、いくつかの矛盾を導入し、最適化されるポリシーは、アルゴリズム設計者が意図しない動作を学習する可能性がある。2つの報酬シグナルの間には先験的な関係はない。報酬整形はRLにおける活発な研究分野であるが、多くの場合、工学的なトリックによって行われている。

報酬整形はRLにおける活発な研究分野であるが、多くの場合、多くの工学的なトリックによって実行される。経験者の実演による教師付き信号からポリシーを学習する場合、パフォーマンス尺度mは、解かれる学習問題にすら現れない。この文脈では、目標は、専門家の政策πeを模倣するために行動空間内の政策π 2を最適化することである（図5で最初に紹介したように）。

（９）式

ここでℓはタスク依存の損失（分類、回帰など）である。状態Sは、インスタンスだけでなく、データを収集するために使用された専門家の方針πeにも条件付きであることを強調した。直観的には、MLモデルがより良く学習すればするほど、すなわち、より良く専門家を模倣すればするほど、学習されたポリシーの最終的な性能は専門家の性能に近づくはずである。いくつかの条件下では、学習されたポリシーのパフォーマンスと専門家のポリシーのパフォーマンスを関連付けることが可能であるが、この側面をカバーすることは本稿の範囲外である。逆に、学習が失敗しても、（代替的な良い決定に遭遇することによって）ポリシーが良いパフォーマンスを示すことがある。実際、代用目的誤差が大きい決定をするとき、実際には、その決定は元のメトリックで良いパフォーマンスを持つことができるのに、学習は完全にペナルティを受けることになる。そのため、パフォーマンス指標を報告することが重要である。例えば、セクション3.2.2でBonami et al.(2018)の研究を調査したが、そこでは著者は混合整数二次問題インスタンスが線形化されるべきか否かを予測するために分類器を歪めた。学習器に使用されるターゲットは、両方の構成で問題インスタンスを解くことによって最適に計算される。単に分類精度を報告するだけでは不十分である。実際、この指標では、誤分類が実行時間に与える影響についての情報は得られない。バイナリ分類では、適切に分類された例でも、2つの構成の実行時間の間に有意な差がないことが起こり得る。この問題を軽減するために、著者らは、実行時間が有意に異ならないカテゴリも導入している（そして、実際の実行時間を報告する）。連続的な拡張は、解答時間の回帰を学習することであろう。しかし、この回帰を学習することは、最終的なアルゴリズムが最適なものを見つけるために、決定の集合を最適化する必要があることを意味する。RLでは、これは価値関数の学習に類似している（セクション2.2参照）。同じ推論を繰り返される決定に適用することは、完全なRL理論でより良く理解される。

P29

4.3 On generalization

セクション4.1では、(6)の確率分布はアクセス不可能であり、有限のデータセットDtrain上の経験的確率分布で置き換える必要があると主張した。最適化問題は

(10)式

セクション2.2で指摘したように、経験的確率分布に対して最適化を行う場合、有限な問題インスタンスに対しては、真の確率分布に関係なく、低い性能尺度になる危険性がある。この場合、訓練性能と真の期待性能との間に不一致が生じるため、汎化誤差が大きくなる（オーバーフィッティング）。この側面を制御するために、汎化性能の推定値に基づいて無限の候補アルゴリズムを比較するための検証集合Dvalidを導入し、選択されたアルゴリズムの汎化性能を推定するためのテスト集合Dtestを用いる。