L'APPRENTISSAGE PROFOND, UNE PUISSANTE ALTERNATIVE POUR LA RECONNAISSANCE D'INTENTION

par

Thibault Duhamel

Mémoire présenté au Département d'informatique
en vue de l'obtention du grade de maître ès sciences (M.Sc.)

FACULTÉ DES SCIENCES

UNIVERSITÉ DE SHERBROOKE

Sherbrooke, Québec, Canada, 20 février 2020

Le 20 février 2020

le jury a accepté le mémoire de Monsieur Thibault Duhamel
dans sa version finale.

Membres du jury

Professeur Froduald Kabanza
Directeur de recherche
Département d'informatique

Professeur Sylvain Giroux
Membre interne
Département d'informatique

Professeur Marc Frappier
Président-rapporteur
Département d'informatique

Sommaire

Ce mémoire s'inscrit dans la lignée d'une avancée de connaissances en reconnaissance d'intention, une discipline de recherche en intelligence artificielle visant à inférer les buts poursuivis par un individu à l'aide d'observations de son comportement. Ce problème, du fait de sa complexité, reste irrésolu dans les domaines réels : les voitures autonomes, les instruments de détection d'intrusion, les conseillers virtuels par messagerie et tant d'autres profiteraient encore actuellement d'une capacité de reconnaissance d'intention.

Longtemps abordé sous l'angle de considérations symboliques spécifiées par des experts humains, le problème commence à être résolu par des approches récentes usant d'algorithmes d'apprentissage dans des contextes simples. Nous nous inspirons ici des progrès de l'apprentissage profond dans des domaines connexes pour en faire usage à des fins de reconnaissance de but à long-terme. Encore sous-exploité pour cette catégorie de problèmes, nous l'avons mis à l'épreuve pour résoudre les problèmes traités dans la littérature et cherchons à améliorer les performances de l'état de l'art.

Pour ce faire, nous présentons trois articles de recherche. Le premier, accepté au workshop PAIR (*Plan, Activity and Intent Recognition*) lors de la conférence AAAI 2018 (*Association for the Advancement of Artificial Intelligence*), propose une comparaison expérimentale entre différentes architectures d'apprentissage profond et les méthodes symboliques de l'état de l'art. Nous montrons de ce fait que nos meilleurs résultats surpassent ces méthodes symboliques dans les domaines considérés. Le deuxième, publié sur arXiv, introduit une méthode pour permettre à un réseau de neurones de généraliser rapidement à plusieurs environnements grâce à une projection des données sur un espace intermédiaire et en s'inspirant des progrès du *few-shot transfer learning*. Enfin, le troisième, soumis à ICAPS 2020 (*International Conference on Automated*

Sommaire

Planning and Scheduling), améliore encore les résultats précédents en fournissant aux réseaux des caractéristiques supplémentaires leur permettant de se projeter dans le futur avec une capacité d'imagination et de résoudre le principal défaut inhérent aux approches symboliques de l'état de l'art, à savoir la dépendance à une représentation approximée de l'environnement.

Mots-clés: Intelligence artificielle ; reconnaissance d'intention ; reconnaissance de plan ; reconnaissance de but ; réseaux de neurones ; apprentissage profond ; transfert d'apprentissage ; connaissances symboliques

Remerciements

Je commence par remercier avant toute chose Froduald Kabanza, mon directeur de recherche, pour avoir accepté de me recevoir dans son laboratoire, concrétisant alors mon projet d'étudier à l'étranger. Au travers de ses retours, commentaires, suggestions et recommandations, il a su m'aiguiller dans le parcours d'apprentissage qu'est une maîtrise pour développer des compétences que je ne possédais pas il y a 16 mois. Je remercie également Marc Frappier et Sylvain Giroux pour l'intérêt qu'ils ont porté à l'évaluation de mon travail.

J'aimerais accorder beaucoup d'importance à ma collègue de recherche Mariane Maynard qui, dans une période difficile pour elle, a su m'accueillir de la meilleure des manières alors que je me trouvais seul à 5 000 kilomètres de tout ce que je connaissais, m'a équipé d'une quantité inestimable de ressources scientifiques et est restée un solide pilier sur lequel j'ai pu compter chaque jour de cette aventure. Je tiens particulièrement à souligner ses qualités qui m'impressionnent encore, à savoir son sérieux, sa persévérance et son implication dans son travail, et ce malgré les aléas qu'implique la recherche.

« Ineffable », voilà le seul mot qui peut exprimer le soutien inconditionnel de ma famille pendant toute cette année, de mon départ à mon retour. Vous avez su écouter et trouver les mots pour me remotiver à chaque période difficile et êtes restés présents à mes côtés à chaque fois que j'étais seul. Je manifeste une gratitude infinie à mes parents, à ma soeur et à mes grands-parents pour cette aide que je n'ai jamais eu à demander et que je ne saurai jamais vous retourner. Ce mémoire est pour vous, sans qui il n'aurait jamais existé.

Abréviations

AAAI Conférence scientifique (*Association for the Advancement of Artificial Intelligence*)

AI Intelligence artificielle (*Artificial Intelligence*)

BFS Recherche en largeur (*Breadth-First Search*)

CNN Réseau de neurones convolutif (*Convolutional Neural Network*)

DL Apprentissage profond (*Deep Learning*)

DNN Réseau de neurones profond (*Deep Neural Network*)

FC Réseau complètement connecté (*Fully Connected*)

HMM Modèle de Markov caché (*Hidden Markov Model*)

ICAPS Conférence scientifique (*International Conference on Automated Planning and Scheduling*)

IRL Apprentissage par renforcement inversé (*Inverse Reinforcement Learning*)

LSTM Réseau à mémoire long terme/court terme (*Long Short-Term Memory*)

MDP Processus de décision de Markov (*Markov Decision Process*)

PAIR Workshop de AAAI (*Plan, Activity and Intent Recognition*)

PDDL Langage pour décrire un domaine de planification (*Planning Domain Description Language*)

SBR Algorithme symbolique de reconnaissance de comportement (*Symbolic Behavior Recognition*)

STDNN Réseau de neurones profond spatiotemporel (*Spatiotemporal Deep Neural Network*)

Table des matières

Liste des figures

Liste des tableaux

Introduction

Les citations de l'introduction redirigent vers la bibliographie à la fin du mémoire (page 106).

Reconnaître l'intention d'autrui a de tout temps été une composante fondamentale de la vie en communauté, humaine comme animale. Utilisée à des fins coopératives ou compétitives, cette capacité est ancrée dans notre comportement intuitif à un tel point qu'elle est devenue systématique. Elle est notamment illustrée par la propension qu'ont les individus à collaborer dans des contextes quotidiens, puisqu'en cette faculté réside une clé de compréhension implicite qui s'étend au delà des mots et de la communication, à savoir une forme de déduction basée sur l'imagination et la projection.

Une intention, au sens le plus empirique du terme, est une notion englobant plusieurs niveaux de complexité. On la définit ici comme représentant l'engagement d'une entité à accomplir une séquence d'actions lui permettant finalement d'atteindre un objectif. Il existe une multitude de dimensions hiérarchiques liées à cette nomenclature : au niveau moteur, par exemple, la simple finalité de lever un bras peut en elle-même s'interpréter comme un but résultant d'une suite d'activations nerveuses inconscientes ou, à l'inverse, comme un mouvement intermédiaire visant à saisir un objet distant. Notre cadre d'étude traite d'une perspective plus reculée encore, celle de la reconnaissance d'intention à long terme, que l'on nomme spécifiquement reconnaissance de but (ou reconnaissance de plan dans le cas ou l'on reconnaît un plan, termes souvent employés de manière abusivement réciproque). Il s'agit là d'un domaine cherchant à inférer l'objectif d'un individu observé (appelé agent) grâce à l'unique observation de son comportement, ce dernier impliquant une certaine capacité de planification de sa

part et supposant une démarche de gestion de ressources disponibles ayant pour but de maximiser un gain dans un futur non-immédiat.

Cette capacité, innée chez l'humain, reste pourtant loin de l'être pour les systèmes artificiels intelligents. Bien que la décennie précédente ait connu des avancées spectaculaires dans cette direction, de nombreux outils profiteraient encore actuellement d'accéder à une telle compétence. On citera non-exhaustivement les voitures autonomes (où compte se rendre ce véhicule ?), les instruments de détection d'intrusion (cet individu cherche t-il à cambrioler une habitation, pirater un serveur ?), les conseillers automatiques par messagerie (cet utilisateur souhaite t-il consulter son compte bancaire ?), la surveillance militaire (cette personne tente t-elle une attaque dans la foule ?), et tant d'autres.

Avant même l'apparition de tels équipements, de nombreux travaux de recherche ont proposé des solutions diverses pour résoudre ce problème de reconnaissance d'intention. La première vague, initiée aux alentours des années 1990, se base sur des connaissances symboliques extraites par un expert humain, qui sont des informations sur l'environnement et les comportements possibles de l'agent, encodés sous la forme de réseaux bayésiens dynamiques [9], de modèles de Markov cachés [8], de grammaires probabilistes [17], de logique de Markov [49] ou de réseaux hiérarchiques de tâches [6]. La deuxième série, remplaçant les bibliothèques de plans par des algorithmes de planification, se basent sur le principe de rationalité, c'est-à-dire l'intuition selon laquelle l'agent essaye de minimiser le coût de ses plans [46, 35].

Néanmoins, les méthodes énoncées ci-dessus sont extrêmement dépendantes de la qualité des modèles qui leur sont fournis. Ainsi, elles sont particulièrement efficaces dans des contextes où les connaissances sont trivialement explicitables, mais deviennent inapplicables à des situations réelles significativement plus complexes, et c'est la raison pour laquelle, aujourd'hui encore, les domaines considérés dans la littérature sont synthétiques. Avec l'arrivée massive et les progrès phénoménaux de l'apprentissage profond, une nouvelle branche de la reconnaissance de plan s'est ouverte : elle propose l'alternative de refaçonner la reconnaissance d'intention en tant que problème de classification. À partir d'un jeu de données, composé de comportements observés, l'approche consiste à optimiser les poids d'un réseau de neurones pour qu'il prédise les buts satisfaits par lesdites observations. Malgré le succès d'une telle

stratégie dans des domaines variés (classification d'images, reconnaissance d'activités à court terme, ...), très peu d'articles s'en sont servis pour la reconnaissance de plan à long terme et, d'autant plus, dans des cadres très particuliers [38, 1, 43]. C'est dans cette perspective que nous introduisons notre travail de recherche.

Nous présentons ce mémoire en trois temps. Dans une première partie, nous proposons d'évaluer la performance de différentes architectures d'apprentissage profond en les confrontant aux approches de l'état de l'art dans leurs domaines respectifs et démontrons de fait que nos structures de réseaux de neurones sont en mesure d'outrepasser les résultats de ces dernières. En deuxième lieu, nous avançons d'un pas vers la généralisation de cette démarche, en étudiant les capacités de transfert d'apprentissage entre des domaines d'applications similaires. Finalement, nous implémentons de nouvelles caractéristiques symboliques (autrement appelées métriques intermédiaires) pour l'apprentissage profond, étudions comment nos modèles profitent de ces informations supplémentaires et vérifions leur robustesse aux connaissances partielles, approximatives et erronées.

Chapitre 1

Le problème de reconnaissance de but

Les citations de ce chapitre redirigent vers la bibliographie à la fin du mémoire (page 106).

Nous commençons tout d'abord par détailler les concepts clés sous-jacents au problème de reconnaissance de but. Après avoir formalisé le problème que nous tâchons de résoudre, nous fournissons un aperçu des différentes techniques les plus impactantes utilisées jusqu'à ce jour pour prédire l'intention d'un agent observé dans son comportement à long-terme, en se concentrant principalement sur les plus récentes.

1.1 Formalisation du problème

Dans ce mémoire, nous nous appuyons sur la formalisation de Sukthankar *et al.* [58]. Nous considérons l'observation d'un seul agent, évoluant dans un environnement neutre et sans interaction avec l'observateur. À partir d'une séquence ordonnée d'observations $O = (o_1, o_2, ..., o_n)$, pouvant provenir directement de capteurs bruts ou d'un pré-traitement en amont, et d'un ensemble de buts possibles G (aussi appelés hypothèses), une méthode de reconnaissance de but retourne une distribution probabiliste $P(G|O)$ sur l'ensemble des buts. Le but prédit est par conséquent celui affichant un

score maximal :

$$g^* = argmax_{g \in G}(P(g|O)) \ . \tag{1.1}$$

Cette définition reste élémentaire, car très généraliste, étant donné que les observations o_i peuvent prendre n'importe quelle forme représentant l'état de l'agent (action instanciée, vidéo, vecteur, ensemble de prédicats, ensemble de valeurs dans un espace intermédiaire, ...). On remarquera d'ailleurs qu'une des premières questions en reconnaissance de but consiste à trouver une représentation adaptée à l'environnement considéré.

La performance des méthodes est typiquement évaluée avec la mesure de précision, qui est le rapport entre le nombre de prédictions correctes et le nombre de prédictions totales. Plusieurs définitions existent dans la littérature pour qualifier une prédiction de correcte. Certaines considèrent, par exemple, qu'une prédiction est correcte si le score maximal de la distribution correspond à celui du vrai but recherché par l'agent. Cependant, nous trouvons cette condition biaisée car elle ne tient pas compte des égalités. Ainsi, une distribution égale à $[0.5, 0.5]$ serait considérée comme correcte alors qu'elle ne départage aucun des deux buts possibles. D'autres valident une prédiction si le score du vrai but se situe dans un top-k des meilleurs scores, ce qui est encore une fois biaisé par la variation arbitraire du k. Nous faisons le choix, dans la suite de ce mémoire, d'utiliser la première définition en y ajoutant un tirage aléatoire en cas d'égalité, pour se rapprocher de situations réelles où un choix doit impérativement être effectué.

1.2 Approches symboliques non probabilistes

Les approches symboliques, à la base de la résolution des problèmes de reconnaissance de but, s'appuient sur des connaissances explicites extraites par des experts humains dans les domaines considérés. Kautz et Allen [26], en 1986, motivent une expansion des recherches basées sur ce paradigme en introduisant une manière de représenter ces connaissances et de réaliser de la reconnaissance de but sur celles-ci. En effet, ils proposent de traduire les actions réalisables dans une structure de graphe

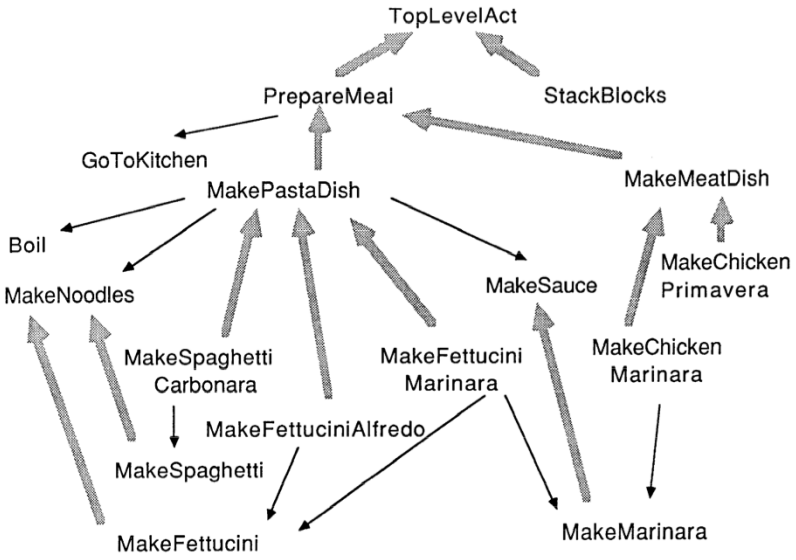

Figure 1.1 – Un exemple de taxonomie, provenant de l'article original [26]

orienté, qu'ils appellent « taxonomie » (la figure 1.1 en donne un exemple). Prédire le but de l'agent se limite alors à calculer la couverte minimale de cette taxonomie à partir de la séquence d'observations.

En 2005, Avrahami-Zilberbrand et Kaminka [3] améliorent la représentation en introduisant des bibliothèques de plans sous la forme d'arbres hiérarchiques temporels, qui indiquent pour chaque action le moment de sa réalisation (figure 1.2). Cela offre un avantage majeur en terme de complexité algorithmique puisque les indications temporelles limitent l'exploration du graphe.

On commence immédiatement à constater les défauts des approches symboliques : elles sont extrêmement dépendantes de la qualité des connaissances extraites. Si une action ou une transition n'est pas instanciée, le comportement de l'agent devient immédiatement inexplicable. D'une même manière, si le comportement de l'agent est irrationnel, sous-optimal ou non-cohérent, l'algorithme d'inférence se voit inefficace. Enfin, les prédictions sont grandement influencées par le modèle décisionnel mathématique choisi pour effectuer les décisions. Par exemple, Kautz et Allen [26] considèrent l'explication formée des hypothèses les plus simples, alors que Avrahami-Zilberbrand et Kaminka [3] produisent toutes les hypothèses, sans les classer.

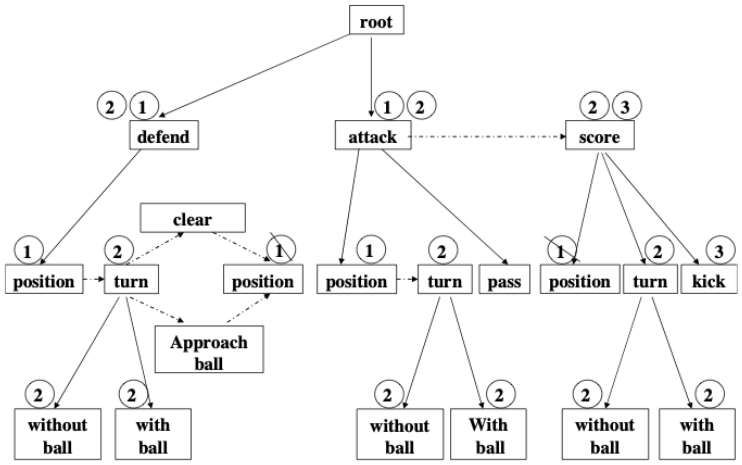

Figure 1.2 – Un exemple de bibliothèque de plans temporelle, provenant de l'article original [3]

1.3 Approches probabilistes

Pour effectuer un choix d'hypothèse parmi un ensemble d'hypothèses cohérentes avec les observations, les approches probabilistes s'intéressent à diverses manières d'attribuer un score à chacune des possibilités ou, en d'autres termes, produisent une distribution de probabilités sur l'ensemble des hypothèses. Si il existe plusieurs hypothèses expliquant les observations, il est maintenant possible de les classer.

Les initiateurs de cette lignée de travaux sont Charniak et Goldman [9], qui expriment les différents états possibles à l'aide de la logique du premier ordre et battissent dynamiquement un réseau bayésien représentant les transitions entre ces états (figure 1.3). Les racines de ce réseau correspondent aux buts réalisables, et leurs probabilités sont calculées en propageant les probabilités conditionnelles.

Sukthankar et Sycara [57] présentent une approche très intéressante en 2005, puisqu'ils essayent de prédire le but d'un individu à partir de capteurs physiques réels détectant ses mouvements. Cependant, ils séparent encore le processus court-terme (reconnaître les actions) et le processus long-terme (reconnaître le but). Pour reconnaître les actions individuelles de l'agent, les auteurs utilisent une machine à vecteurs de support pour classifier ses mouvements dans un contexte de surveillance militaire

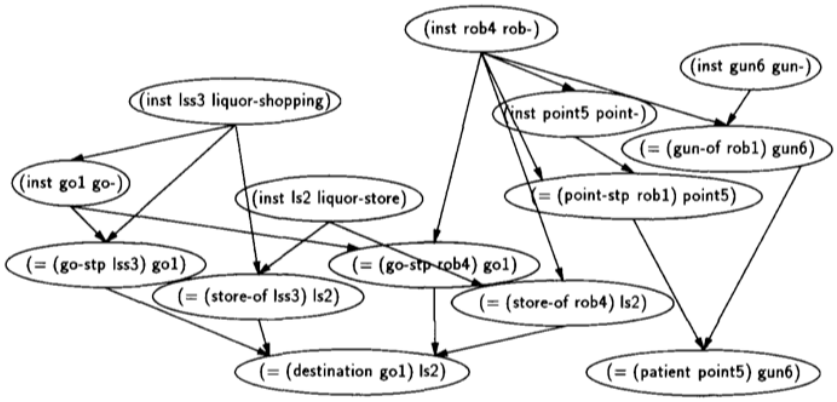

Figure 1.3 – Un exemple de réseau bayésien, provenant de l'article original [9]

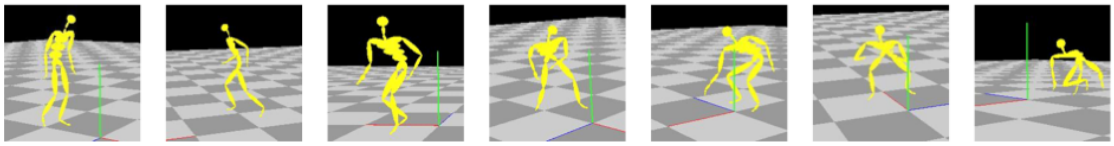

Figure 1.4 – Les actions de l'agent dans l'article original [57]

(les différentes actions sont affichées sur la figure 1.4) et un modèle de Markov caché pour discerner les erreurs de classification (en considérant les probabilités d'observer deux actions successives). Enfin, pour prédire le but d'un agent, la séquence d'actions obtenue est recherchée dans une bibliothèque de plans exhaustive, dans laquelle un coût est affecté à chaque transition. Le but le plus probable est alors celui qui explique les observations et qui minimise le coût total. À noter que l'on retrouvera également cette intuition de minimisation du coût dans la section suivante sur les approches génératives.

Geib et Goldman [17] étudient l'application des arbres de grammaires hiérarchiques dans le cadre d'une détection d'intrusion d'un système informatique, construisant ainsi une méthode qu'ils baptisent *PHATT (Probabilistic Hostile Agent Task Tracker)*. Pour ce faire, les plans réalisables par l'agent sont représentés à l'aide d'arbres logiques (figure 1.5). On y observe des hypothèses à la racine, des actions pour réaliser ces hypothèses et des règles entre ces actions. Les arcs de cercle dénotent une relation

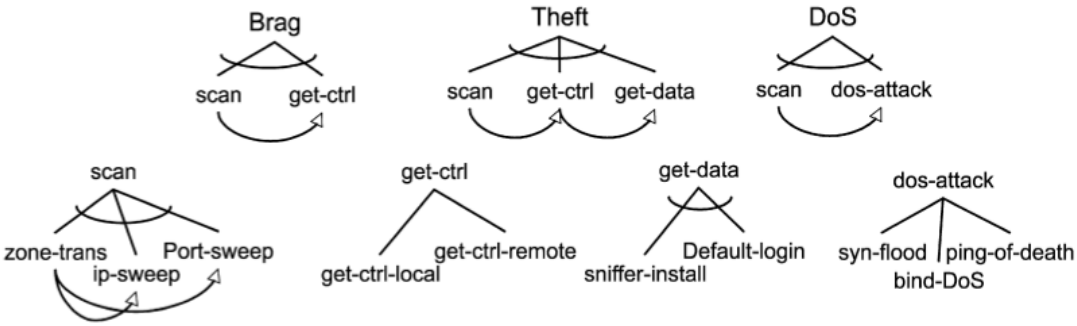

Figure 1.5 – Un exemple de grammaires hiérarchiques, provenant de l'article original [17]

« ET », ce qui signifie que chaque action fille doit être observée. Dans ce cas, une flèche indique l'ordre dans lequel les actions doivent être réalisées. L'absence d'un arc de cercle représente une relation « OU », ce qui indique que n'importe quelle action fille peut être observée. L'algorithme de reconnaissance de but associé est très attrayant car il permet de sauvegarder l'état des différentes hypothèses à chaque nouvelle observation, ce qui signifie qu'il est possible de reconnaître plusieurs buts exécutés en parallèle.

Bien que beaucoup plus puissantes que les approches purement symboliques, ces méthodes affichent toujours le même défaut quant à l'extraction des connaissances complètes de l'environnement. De plus, elles sont aussi sensibles au choix de la formulation mathématique et des paramètres intervenant dans le calcul des probabilités. La précision est donc intrinsèquement liée aux approximations créés par ces métriques. Par exemple, la méthode de Sukthankar et Sycara [57] est extrêmement dépendante des coûts manuellement affectés à chaque transition. De la même manière, *PHATT* [17] nécessite de connaître à l'avance les différentes attaques possibles, ce qui n'est pas possible en sécurité informatique.

1.4 Approches génératives

Ce paradigme, introduit par Baker *et al.* [4] et Ramírez et Geffner [45] en 2009, suppose l'application du principe de rationalité sur le comportement de l'agent. Cela

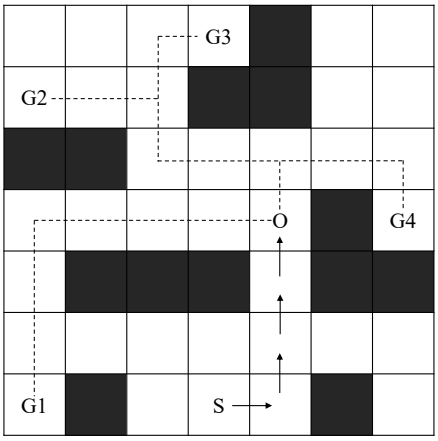

Figure 1.6 – Exemple de trajectoire d'un agent dans une grille avec 4 buts

signifie que, en fonction de ses connaissances sur l'environnement, celui-ci va toujours chercher à accomplir son objectif de manière optimale (ou du moins, d'une manière la plus optimale possible). Cette perspective revient alors à se mettre à la place de l'agent et à inverser son processus de planification. Plus il dévie d'un comportement optimal pour un but donné, moins ce but est probable. Intuitivement, sur la figure 1.6, on peut estimer que l'agent ne se dirige pas vers G1 car il n'est pas sur un chemin optimal vers ce but, puisqu'il aurait à revenir en arrière pour y arriver. Ainsi, au lieu d'expliciter exhaustivement les buts et les plans possiblement réalisables, on a simplement besoin d'un domaine de planification et d'un planificateur associé (ce qui reste néanmoins difficile à extraire).

Formellement, Ramírez et Geffner [46] définissent la vraisemblance d'un but avec la formule de Boltzmann suivante :

$$P(O|g) = \frac{e^{-\beta\Delta(s,g,O)}}{1 + e^{-\beta\Delta(s,g,O)}} \ , \tag{1.2}$$

où β est un paramètre contrôlant la certitude de la prédiction et Δ est la différence de coût définie comme suit :

$$\Delta(s, g, O) = c^*(s, g, O) - c^*(s, g, \neg O) \ , \tag{1.3}$$

avec s étant l'état de départ, $c^*(s, g, O)$ le coût d'un plan optimal entre s et g respec-

tant les observations (i.e. contenant O de manière monotone) et $c^*(s, g, \neg O)$ le coût d'un plan optimal entre s et g ne respectant pas O.

À partir de $P(O|g)$, il est possible d'obtenir $P(g|O)$ à l'aide de la relation de Bayes $P(g|O) = \alpha P(O|g)P(g)$, en supposant P(g) uniforme dans la majorité des problèmes.

Suite à cela, Sohrabi *et al.* [53] proposent un amélioration permettant de gérer les observations bruitées et les comportements relativement sous-optimaux. Pour ce faire, une nouvelle formule de coût est proposée :

$$V(s, g, O) = c(s, g, O) + b_1 M(s, g, O) + b_2 N(s, g, O) \ , \tag{1.4}$$

où $M(s, g, O)$ est le nombre d'observations manquantes et $N(s, g, O)$ est le nombre d'observations bruitées. Pour gérer les comportements sous-optimaux, $c(s, g, O)$ dépend du coût des k meilleurs plans (*top-k*), où k varie arbitrairement (fixé à 1000 dans l'article, ce qui augmente considérablement le temps d'exécution).

Ces approches, bien qu'efficaces, sont lourdes à appliquer en pratique car elles requièrent respectivement $2|G|$ et $k|G|$ appels à un algorithme de planification. De nombreux papiers ont tenté de résoudre ce problème en incorporant des approximations dans le processus de reconnaissance de but pour réduire la durée des prédictions, mais au détriment de la précision.

Vered et Kaminka [61] réduisent le nombre d'appels à un planificateur et prouvent qu'ils sont capables de reconnaître le but d'un agent en temps réel (*online*), c'est-à-dire au fur et à mesure que son comportement est observé. Pour cela, ils définissent deux nouvelles règles pour chaque nouvelle observation. Si celle-ci change le classement des buts (ce qui est évalué avec une heuristique), alors il est nécessaire de recalculer un plan pour chaque but. Sinon, les plans calculés à l'instant précédent sont conservés. De plus, si un but devient improbable (selon un seuil géométrique arbitraire), il est retiré de la liste des buts possibles. Les auteurs annoncent un nombre d'appels minimal de $|G|$ et un nombre maximal de $|G|(|O|+1)$, ce qui reste tout de même élevé. Le principal défaut de cette méthode, au final, réside dans la perte de qualité des prédictions. En effet, les résultats expérimentaux démontrent une perte de 50% de précision pour un gain de 170% en rapidité d'exécution. On peut également ajouter que les heuristiques définies se basent sur des intuitions géométriques, valables uniquement dans des contextes où elles s'appliquent.

Pereira *et al.* [42] définissent une heuristique intéressante se basant sur des jalons (*landmarks*). Un jalon est une observation cruciale pour se rendre à un but donné ; en d'autre termes, il est impossible de se rendre à ce but sans avoir traversé tous les jalons nécessaires. À partir de cette définition, les auteurs considèrent le ratio de complétion entre le nombre de jalons observés et le nombre total de jalons, pour chaque but, en estimant qu'un but est plus probable si ce ratio est élevé. Bien que plus efficace que Ramírez et Geffner [46] en terme de complexité et de rapidité d'exécution, la méthode affiche encore des précisions inférieures. On note aussi que les résultats sont fortement liés à qualité des jalons extraits à partir de connaissances sur l'environnement, ce qui ne résout pas le problème des approches symboliques.

Masters et Sardiña [35] amènent une simplification efficace de la formule de Ramírez et Geffner [46] :

$$\Delta(s, g, n) = c^*(n, g) - c^*(s, g) \ , \tag{1.5}$$

où n est le dernier état de l'agent observé. De ce fait, les prédictions ne dépendent plus de la séquence d'observations. Il est dès lors possible de calculer des cartes de coûts à l'avance et d'y accéder en temps réel avec une complexité de $O(1)$. Les auteurs démontrent qu'il n'y a pas de perte de précision dans les domaines où la dernière observation contient suffisamment d'informations pour prédire le but de l'agent.

1.5 Approches par apprentissage

De nouvelles approches ont appliqué le concept d'apprentissage pour discerner automatiquement des informations essentielles pour la reconnaissance de but à partir d'exemples de comportements.

Bisson *et al.* [6], en 2015, se basent sur des arbres hiérarchiques (HTN) pour instancier les différents plans réalisables (voir figure 1.7), où l'on retrouve les liaisons « ET » (ordonnés temporellement avec des flèches) et « OU », mais en y ajoutant des poids pour contrôler l'incertitude. En effet, pour un but donné (à la racine d'un arbre), les auteurs définissent une fonction récursive paramétrée h_θ telle que :

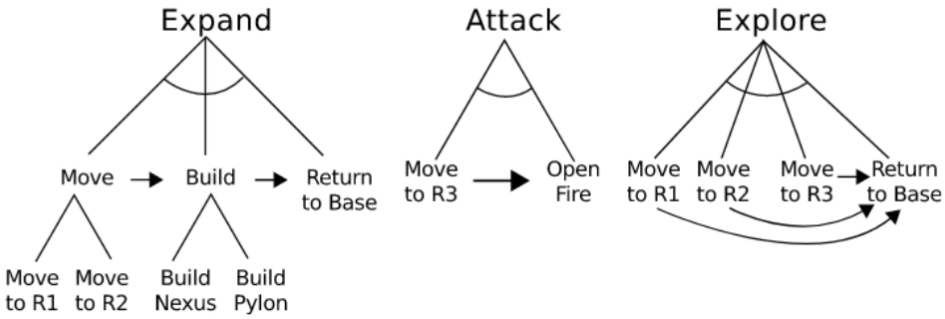

Figure 1.7 – Un exemple d'arbre hiérarchique pour le jeu de Starcraft, provenant de l'article original [6]

$$h_\theta(x) = g\Big[b + \frac{1}{w}\sum_{i=1}^{w} W h_\theta(x_i) + \frac{1}{u}\sum_{i=1}^{u} U h_\theta(x_i) + \frac{1}{v}\sum_{i=1}^{v} V c_{i,\theta}(x)\Big] \,, \tag{1.6}$$

où g est une fonction d'activation (e.g sigmoïde), b est un vecteur de biais, w est le nombre de noeuds fils observés pour x, u est le nombre de noeuds fils non-observés pour x, v est le nombre de contraintes ordonnées sous x, W est la matrice de poids pour les noeuds fils observés, U est la matrice de poids pour les noeuds fils non-observés, V est la matrice de poids pour les contraintes ordonnées, et $c_{i,\theta}(x)$ est la concaténation des valeurs de h_θ pour la partie gauche et droite d'une contrainte donnée, $c_{i,\theta}(x) = [h_\theta(c_{i->gauche}), h_\theta(c_{i->droite})]$.

Les poids de cette fonction récursive sont alors classiquement entraînés par propagation suivant une descente de gradient stochastique, en minimisant la fonction de log-vraisemblance pour chaque exemple. Lors d'une prédiction, un score est calculé récursivement pour chaque but en fonction des poids entraînés et des observations observées. Les résultats démontrent une amélioration des performances dans certains domaines, mais l'approche nécessite toujours d'expliciter une bibliothèque de plans manuellement, bien que possiblement imparfaite.

Granada *et al.* [19], en 2017, proposent une méthode hybride séparant l'aspect bas-niveau, pour reconnaître les actions dans une vidéo, de l'aspect haut-niveau, pour reconnaître le but à partir de ces actions. Dans la première partie, un réseau à convolu-

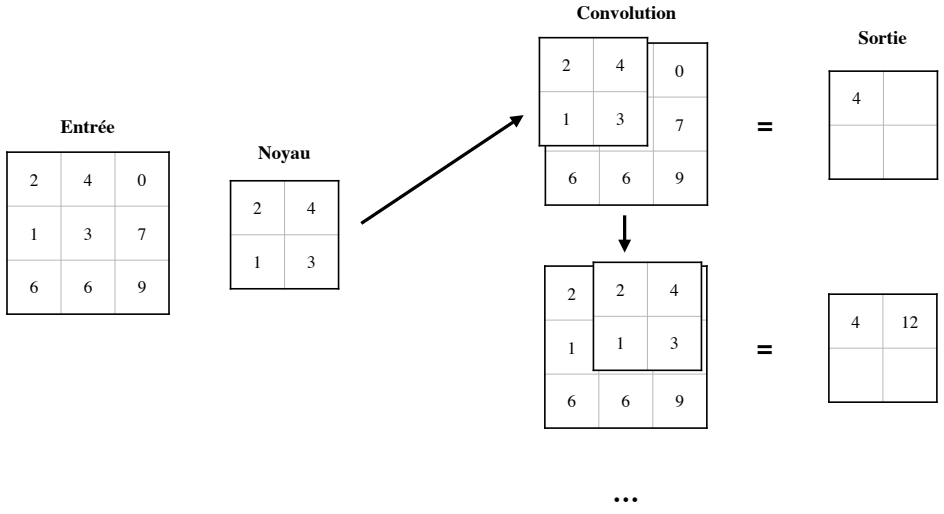

Figure 1.8 – Un exemple de réseau à convolution

tion (*CNN*) est utilisé, puisque particulièrement souhaitable pour traiter des images à deux dimensions spatiales. Un réseau à convolution traite une matrice en entrée selon la formule suivante :

$$h_{x',y'} = \sigma(\theta * a) = \sigma(\sum_{x,y=0}^{M,N} \theta_{x,y} a_{x'-x,y'-y}) \ , \tag{1.7}$$

où h est la matrice de sortie, σ une fonction d'activation, θ est le noyau de taille $N \times M$, $*$ est l'opération de convolution et a est une fenêtre de taille $N \times M$ sur l'entrée. La figure 1.8 illustre le procédé.

À partir des actions ainsi identifiées, la reconnaissance de but en tant que telle est effectuée avec un algorithme SBR (*Symbolic Behavior Recognition*) qui compare les actions observées avec une bibliothèque de plans. Bien qu'appliquée dans un domaine réel (préparer des repas dans une cuisine), la méthode sépare encore la reconnaissance de but symbolique de l'apprentissage.

Récemment, Pereira *et al.* [43] ont introduit une manière d'apprendre automatiquement un modèle de l'environnement à partir des observations, grâce à un réseau dense complètement connecté (*FC*), formé d'un enchaînement de plusieurs transfor-

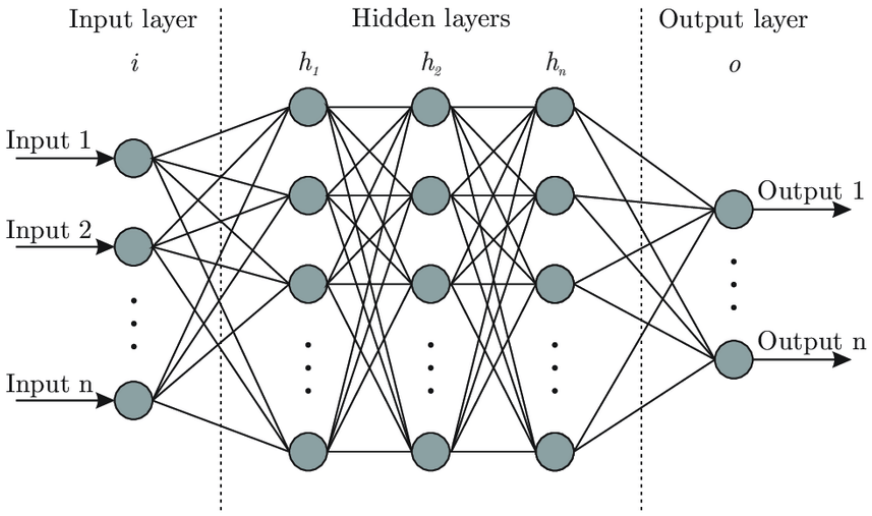

Figure 1.9 – Un exemple réseau dense (source : [7])

mations linéaires, entrecoupées d'activations σ :

$$h_{i+1} = \sigma(\theta_i.h_i + b) \ , \tag{1.8}$$

où b est un vecteur de biais. La figure 1.9 est un exemple de représentation de cette catégorie de réseaux, où les états intermédiaires cachés sont illustrés par des noeuds et les poids sont représentés par des transitions entre ces noeuds. Dans le cas de Pereira *et al.* [43], le réseau prend en entrée la représentation d'un état et d'une action pour calculer l'état résultant. Une fois qu'une représentation du modèle est apprise, la reconnaissance de but s'effectue simplement avec une comparaison symbolique basée sur les coûts des chemins. L'approche est intéressante car elle permet de s'affranchir de connaissances à extraire manuellement. Cependant, il n'est pas trivial de pouvoir contrôler la qualité du modèle appris. De plus, l'hypothèse de rationalité des agents est encore une fois prise en compte, ce qui n'est pas toujours le cas avec les comportements humains.

La formalisation du problème de reconnaissance de but se rapproche intrinsèquement d'un problème de classification. La pure utilisation de l'apprentissage profond semble alors attrayante pour le résoudre mais n'a fait l'objet que d'un nombre limité

d'études dans des situations spécifiques.

Supposons qu'il existe un ensemble $\mathcal{O} = (O_i)_{1 \leq i \leq N}$ de séquences d'observations et un ensemble $\mathcal{G} = (g_i)_{1 \leq i \leq N}$ de buts "étiquettes" associés à ces séquences. Alors, on peut exhiber une fonction f qui associe le vrai but de l'agent à chacune des séquences, telle que :

$$\forall i \in [1, N], f(O_i) = g_i \ . \tag{1.9}$$

En pratique, f est évidemment inconnue, ce qui nous amène à utiliser une fonction f' approximant f à l'aide d'un ensemble de paramètres θ. Le rôle de l'apprentissage profond sera dès lors de minimiser l'erreur (l'écart) entre f' et f :

$$\theta = argmin_\theta \sum_{i=1}^{N} L(f'(O_i, \theta), f(O_i)) = argmin_\theta \sum_{i=1}^{N} L(f'(O_i, \theta), g_i) \ , \tag{1.10}$$

où L est une fonction de perte, nulle lorsque $f'(O_i, \theta) = f(O_i)$.

Classiquement, les paramètres θ sont optimisés à l'aide d'une descente de gradient itérative :

$$\theta_{i+1} = \theta_i - \alpha_i \frac{dL}{d\theta_i} \ , \tag{1.11}$$

où α_i est le taux d'apprentissage contrôlant la vitesse et la qualité de l'optimisation.

Min *et al.* [37, 38] sont les premiers à avoir utilisé une solution d'apprentissage profond complète (c'est-à-dire de bout en bout, *end-to-end*) pour apprendre à discerner des motifs de reconnaissance de but dans un ensemble d'observations étiquetées, pour le jeu vidéo *Crystal Island*. Pour cela, les auteurs exploitent la temporalité des séquences d'observations avec un réseau récurrent à mémoire long terme/court terme (*LSTM*), dont l'expression à un instant t est la suivante :

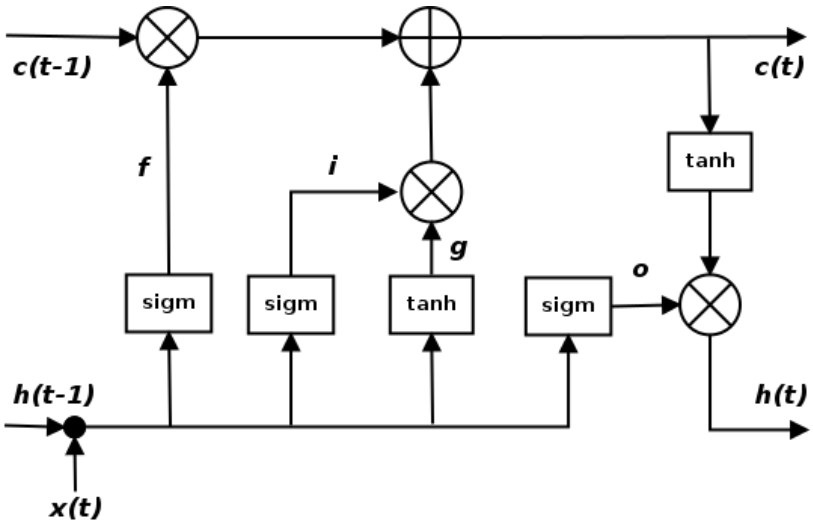

Figure 1.10 – Une cellule d'un LSTM, à un instant t

$$\begin{cases} F_t = \sigma(\theta_F x_t + \phi_F h_t + b_f) \ , \\ I_t = \sigma(\theta_I x_t + \phi_I h_t + b_I) \ , \\ O_t = \sigma(\theta_O x_t + \phi_O h_t + b_O) \ , \\ c_t = F_t \bigodot c_{t-1} + I_t \bigodot tanh(\theta_c x_t + \phi_c h_{t-1} + b_c) \ , \\ h_t = O_t \bigodot tanh(c_t) \ , \\ o_t = \gamma(\theta_o h_t + b_o) \ , \end{cases} \tag{1.12}$$

où F (*forget gate*), I (*input gate*), O (*output gate*), h et c sont des états cachés initialisés à $h_0 = 0, c_0 = 0$ qui contrôlent la propagation des informations entre deux instants successifs, σ est la fonction sigmoïde, θ, ϕ et b sont les paramètres à optimiser, $\bigodot$ est le produit terme à terme, et γ est la fonction d'activation de sortie. Une représentation est donnée en figure 1.10.

L'utilisation d'une telle méthode présente de nombreux avantages et a en partie inspiré nos travaux. En effet, elle ne requiert pas de connaissances préalables sur l'environnement, ce qui signifie que seuls les comportements entrent en compte pour effectuer des prédictions. Il n'y a également aucune hypothèse émise quant à la rationalité des agents observés. Cependant, on remarque que les auteurs effectuent un

traitement des données en amont de l'utilisation du réseau, et c'est la raison pour laquelle la méthode est valable uniquement dans le contexte pour laquelle elle est construite.

Enfin, Amado *et al.* [1] construisent une méthode complète d'apprentissage profond dans des contextes de jeux simples (taquin 8 pièces, tours de Hanoï, ...), séparée en trois parties. Premièrement, un encodeur (réseau dense) apprend à transformer chaque état du domaine en matrice unique dans un espace intermédiaire de représentations latentes. Ensuite, une fois les séquences d'observations transformées, un simple *LSTM* produit en sortie une représentation du but prédit dans le même espace latent. Enfin, un décodeur (réseau dense également) inverse le processus d'encodage pour retrouver le but à partir de sa représentation latente. Cette méthode semble attrayante mais n'a été testée que dans des domaines simples et arbitrairement transformés en problème de reconnaissance de but (par exemple, dans le jeu du taquin, des buts sont générés aléatoirement, ce qui n'a pas vraiment de sens dans ce cadre).

1.6 Remarque : Apprentissage par renforcement inverse

La reconnaissance de but, une fois transformée en problème de classification, affiche de nombreuses similitudes avec l'apprentissage par renforcement inverse (*Inverse Reinforcement Learning, IRL*) [40]. L'objectif de ce dernier consiste à trouver une fonction de récompense (c'est-à-dire une fonction qui indique à quel degré il est souhaitable de se trouver dans un état donné) à partir d'exemples de comportements. Le lien entre les deux problèmes est assez direct, bien que ceux-ci soient différents par définition. En effet, on peut discerner une relation entre l'observation d'un agent minimisant des coûts d'un côté et maximisant une récompense de l'autre. Cependant, dans le cadre de l'*IRL*, on ne cherche pas à prédire le but de l'agent mais simplement à le guider vers celui-ci, en supposant que son comportement est optimal, ce qui n'est pas le cas pour la reconnaissance de but. Néanmoins, on remarque que ce paradigme pourrait être adapté pour estimer les coûts des transitions à partir des observations au lieu de les spécifier manuellement, ou mieux encore, pour prédire les buts à partir

des récompenses.

Rhinehart et Kitani [48] estiment être les premiers à proposer de reconnaître les activités (sous-buts intermédiaires) et les buts d'un agent humain en utilisant l'*IRL*. Bien que l'approche travaille avec des vidéos réelles (en mode de perception première personne) dans un appartement, l'algorithme d'inférence n'utilise pas directement les données brutes, mais un processus de décision de Markov (*Markov Decision Process, MDP*) supposé extrait préalablement à partir des images. Les états possibles pour l'agent intègrent trois quantités : sa position dans l'espace (x, y, z), les derniers buts atteints, et les objets en sa possession. Les buts sont des états qui correspondent à un arrêt de l'agent (c'est-à-dire une vitesse nulle pendant un certain Δt), et on distingue deux types d'actions, que sont les déplacements dans l'espace et les interactions avec les objets (saisir/poser). Il est important de réaliser que cette représentation implique de connaître tous les lieux et tous les objets à la disposition de l'agent à l'avance et, par extension, toutes ses actions. Quant aux transitions, elles sont identifiées en temps réel sous la forme de tuples (*état en entrée*, *action*, *état en sortie*). Une fois un tel modèle établi, la fonction de récompense est apprise classiquement suivant une descente de gradient. Pour inférer le but de l'agent à partir de ces récompenses, les auteurs emploient la distribution suivante :

$$P(g|s_0, ..., s_t) = \alpha P(g) e^{V(s_t,g)-V(s_0,g)} , \tag{1.13}$$

où α est un facteur de normalisation, $P(g)$ est une distribution supposée uniforme à priori, $V(s_t, g)$ est la fonction d'utilité (*value function*, calculée à partir de la fonction de récompense) du but g en considérant le chemin terminant par le dernier état observé s_t et $V(s_0, g)$ est la fonction d'utilité du but g en considérant l'état initial s_0 seulement. Il est extrêmement intéressant de remarquer que, en définitive, il s'agit de la même intuition que Ramírez et Geffner [46] et Masters et Sardiña [35], s'appuyant sur l'hypothèse de rationalité.

Cette méthode est attirante car elle formalise des notions reliant *IRL* et reconnaissance de but. Néanmoins, on y retrouve les défauts énoncés dans les sections précédentes. Bien qu'introduite comme fonctionnant sur des données réelles, elle ne les utilise pas directement et suppose toujours des connaissances (moindres, certes) sur l'environnement, comme les objets ou les lieux disponibles. De plus, l'hypothèse

de rationalité est encore appliquée au travers d'une formule mathématiques explicitée par un expert. Enfin, les résultats sont mitigés ; ils surpassent les approches de la littérature pour le domaine considéré mais ne dépassent pas 50% de précision, en moyenne sur les séquences, pour 5 buts.

Xu *et al.* [70] proposent ouvertement de fusionner la notion d'intention et de récompense pour apprendre à un agent à réaliser des tâches. Bien que différent d'un problème d'inférence, les principes de méta-apprentissage et de *few-shot learning* (apprentissage avec peu d'exemples) y sont appliqués, ce qui fait écho au chapitre 3 de ce mémoire, supposant que des connaissances sont partagées entre des tâches similaires. L'approche de méta-IRL présentée consiste à apprendre une fonction de récompense paramétrée avec des poids θ, non pas optimale pour une tâche donnée, mais facilement adaptable à de nouveaux poids Θ_τ pour une nouvelle tâche τ jamais vue auparavant. Pour cela, les auteurs considèrent plusieurs ensembles distincts de données. Le premier, appelé ensemble de méta-entraînement (*meta-training set*, $\{x_i\}_{1\leq i\leq N}$), est séparé en exemples d'entraînement pour chaque tâche τ_i. Il est utilisé pour optimiser les poids θ sur plusieurs tâches. Le deuxième, ensemble de méta-test (*meta-testing set*, $\{y_j\}_{1\leq j\leq M}$), est utilisé pour quantifier la performance des poids θ sur de nouvelles tâches. Enfin, pour l'évaluation, l'ensemble d'entraînement (*training set*, τ_{train}) contient des exemples pour une seule tâche, jamais vue auparavant, sur laquelle les poids θ sont adaptés vers Θ_τ. L'ensemble de test (*testing set*, τ_{test}) est alors utilisé pour évaluer les performances des nouveaux poids sur la même tâche que τ_{train}.

Le processus de méta-apprentissage revient alors à une effectuer une double optimisation :

$$\theta = \text{argmin}_\theta \sum_{i=1}^{N} L(y_i, \Theta_{\tau_i}) = \text{argmin}_\theta \sum_{i=1}^{N} L(y_i, \theta - \alpha \sum_{j=1}^{M} \nabla_\theta L(x_j, \theta)) \ , \tag{1.14}$$

où L est la fonction de log-vraisemblance et α est le taux de méta-apprentissage.

Les expérimentations sont effectuées dans deux domaines synthétiques (un mini-jeu 2D et un simulateur 3D) avec des tâches simples (se déplacer, prendre/poser des objets) et démontrent que l'approche s'adapte beaucoup mieux avec peu d'exemples,

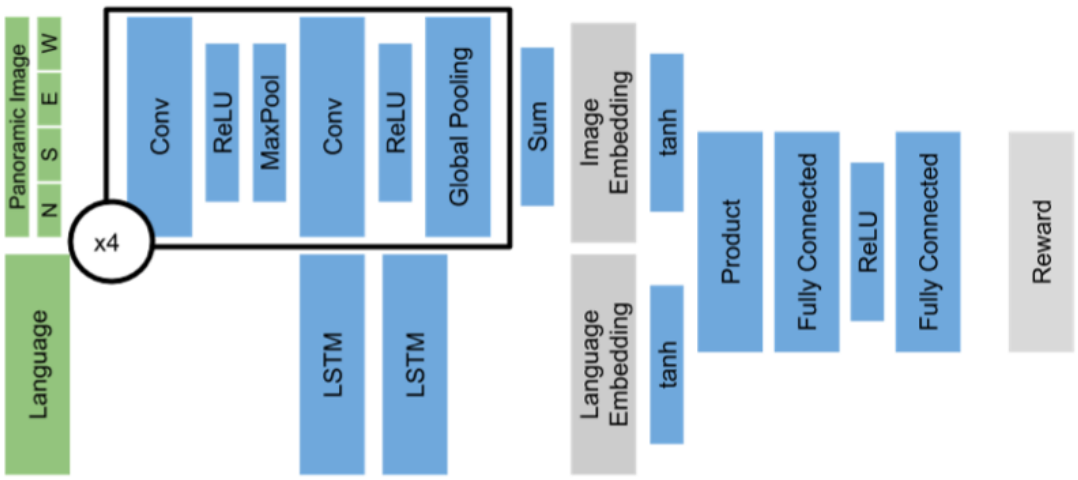

Figure 1.11 – Représentation de l'architecture de la fonction de récompense, provenant de l'article original [15]

en comparaison à différentes baselines. Cela rejoint nos résultats et conclusions dans le chapitre 3.

Fu *et al.* [15] utilisent une architecture d'apprentissage profond pour modéliser une fonction de récompense à partir d'une instruction en langage naturel. La méthode ne reconnaît pas l'intention en soi, mais associe plutôt automatiquement une consigne (objectif) haut-niveau à une séquence d'actions. Pour cela, l'agent évolue dans un environnement synthétique grâce à deux entrées : des images de 4 capteurs virtuels (Nord, Sud, Est, Ouest) et une instruction (comme *Aller à X* ou *Déplacer l'objet X vers Y*). Les images sont traitées avec une série de réseaux à convolution et l'instruction est décomposée avec des *LSTMs*. L'ensemble est ensuite concaténé et donné à un réseau dense. La figure 1.11 illustre l'architecture.

Même si le problème résolu par cet article n'est pas celui de la reconnaissance de but, la question que nous posons dans ce mémoire s'y trouve être liée et, pourtant, exactement opposée : serait-il possible de créer un processus inverse pour retrouver l'instruction (intention) d'origine à partir du comportement de l'agent ?

Chapitre 2

Comparaison entre les approches basées sur les coûts et l'apprentissage profond pour la reconnaissance de but

Les citations de ce chapitre redirigent vers les références à la page 42.

Résumé

Cet article présente une comparaison entre deux méthodes de l'état de l'art [22, 17] et différentes architectures de réseaux de neurones pour la reconnaissance de but dans divers environnements synthétiques considérés. Les premières approches, symboliques, sont basées sur le principe de rationalité et supposent que l'agent observé a un comportement proche d'être optimal. De cette manière, les prédictions réalisées ne considèrent qu'un seul indicateur : le coût des séquences d'actions. Ainsi, plus l'agent dévie d'une attitude optimale vers un but, moins celui-ci est probable. En revanche, ces techniques sont lourdes à mettre en place dans des contextes complexes (comme le monde réel) car elles nécessitent un planificateur

fastidieux et une modélisation exhaustive du domaine.

En opposition à cela, l'apprentissage profond semble être une alternative puissante permettant de dépasser les limites précédemment énoncées. En effet, sa capacité à extraire des informations pertinentes de données brutes sans l'aide de règles pré-établies répond parfaitement au problème principal que nous essayons de résoudre ici. Nous avons expérimenté différentes architectures de réseaux profonds (*FC*, *CNN*, *LSTM*, ...) en adaptant la structure des données traitées (composantes spatiales, temporelles, ...) et montré que nos meilleurs résultats dépassaient ceux des algorithmes symboliques dans les domaines étudiés. Cependant, il y manque encore une possibilité de généralisation, ce qui fera l'objet de nos futurs articles.

Commentaires

Une version antérieure a été publiée à PAIR (*Plan, Activity and Intent Recognition*) en 2019 [1]. La version intégrée dans ce mémoire a été corrigée et améliorée. Elle correspond à la version publiée sur arXiv [2].

Thibault Duhamel et Mariane Maynard ont élaboré ce projet dans le cadre de leur maîtrise en informatique. Thibault a mené le travail de recherche et réalisé les expérimentations. Mariane a dirigé l'écriture de l'article tout en l'assistant et le conseillant. Froduald Kabanza a supervisé l'ensemble du projet et a amené des commentaires, propositions et suggestions sur les travaux et la rédaction.

1. http://www.planrec.org/PAIR/Resource_files/PAIR19papers.zip
2. https://arxiv.org/abs/1911.10074

Cost-Based Plan Recognition Meets Deep Learning [3]

Mariane Maynard, Thibault Duhamel, Froduald Kabanza
Département d'informatique
Université de Sherbrooke
Sherbrooke, Québec (Canada) J1K 2R1
mariane.maynard@usherbrooke.ca,
thibault.duhamel@usherbrooke.ca,
froduald.kabanza@usherbrooke.ca

Abstract

The ability to observe the effects of actions performed by others and to infer their intent, most likely goals, or course of action, is known as a plan or intention recognition cognitive capability and has long been one of the fundamental research challenges in AI. Deep learning has recently been making significant inroads on various pattern recognition problems, except for intention recognition. While extensively explored since the seventies, the problem remains unsolved for most interesting cases in various areas, ranging from natural language understanding to human behavior understanding based on video feeds. This paper compares symbolic inverse planning, one of the most investigated approaches to goal recognition, to deep learning using CNN and LTSM neural network architectures, on five synthetic benchmarks often used in the literature. The results show that the deep learning approach achieves better goal-prediction accuracy and timeliness than the symbolic cost-based plan recognizer in these domains. Although preliminary, these results point to interesting future research avenues.

3. An earlier version of this paper was published to PAIR (AAAI 2019 workshop).

2.1 Introduction

The ability to infer the intention of others, also known as goal, plan, or activity recognition, is central to human cognition and presents a wide range of application opportunities in many areas. Human behavior is often the result of conscious and unconscious cognitive planning processes [24, 3]. Therefore, to infer the intention of other people interacting with us, our brain is somehow able to predict what might be their goals or plans based on observations of clues from their actions. This capability is central to interact smoothly with people, to avoid danger in many situations, and to understand situations unfolding before us, such as predicting the behaviors of pedestrians when driving. Not surprisingly, there is intense research on intention recognition on many AI problems ranging from natural language understanding [33] and human-machine interaction [7] to autonomous vehicles [31] and security monitoring.

Intention recognition is part of the more general problem of pattern recognition, with the critical nuance that it deals with goal-oriented patterns. Deep learning has been making significant inroads in recognizing patterns in general. Latest computer vision algorithms are now able to identify simple human behaviors involving short sequences of actions from videos, such as talking, drumming, skydiving, walking, and so on [25, 14, 35]. However, recognizing behaviors involving longer goal-oriented sequences of actions and produced by elaborate planning processes is another challenge yet barely tackled by end-to-end deep learning solutions [18, 19, 1].

For a long time, various symbolic inference paradigms have been experimented to try to infer the intention from observations based upon handcrafted models, using probabilistic inference frameworks such as HMM [5], Dynamic Bayesian Networks [6], Markov logic [23], probabilistic grammar parsing [11], cost-based goal recognition [22, 17], etc. These approaches require that human experts provide models of behaviors (e.g., domain theories or plan libraries [28]), serving as input to inference engines. However, like vision, language understanding, and other perception tasks, intent recognition is difficult to express in a model, and this often results in a biased or utterly inaccurate definition of the domain for the inference engine. The appeal of representational learning is indeed the ability to extract modeling features, otherwise

difficult to explain for an expert, from data.

In this paper, we show that familiar deep neural network architectures, namely dense, convolutional, and LTSM networks, can perform well on intention recognition problems in navigation domains compared to symbolic cost-based goal recognition algorithms considered as state of the art on this problem [22, 17]. In this domain, we study the case of an agent (the observee) navigating in an environment, for whom the map is known *a priori*, where several points of interest are their potential destinations. It is a synthetic benchmark, with some simplifications, but is a step towards solutions that will work eventually in more realistic environments.

While preliminary, results show that deep learning gives better and quicker goal-prediction accuracy than the state-of-the-art symbolic method. Comparisons on other academic benchmarks often used to evaluate symbolic plan recognizers also suggest that deep neural networks offer competitive performance. It seems that even a simple dense structure can learn abstractions underlying sequential decisions conveyed in the observed patterns of a goal-directed agent enough to outperform a cost-based approach. Before these experiments, we expected the latter to perform better since it is inherently tailored to deal with consecutive decisions. These surprising results raise exciting avenues of investigation that we discuss in the paper.

The rest of the paper follows with a brief review of the most related work, background, experiment methodology, experiment results, and conclusion.

2.2 Related Work

A few approaches combine deep learning and symbolic inference in different ways. For example, Granada *et al.* [13] use a deep neural network to recognize individual actions of an actor cooking recipes in a kitchen, and then use a symbolic algorithm, SBR, to infer the goal underlying an observed sequence of actions. This approach also requires a handcrafted model (plan library) representing abstractions of potential plans the agent could execute. Moreover, no mechanism are allowing the handcrafted plan library to adapt to the classification errors made by the neural network recognizing individual actions.

The procedure in Bisson *et al.* [4] also makes use of a symbolic algorithm, which

requires as input a sequence of observations of actions performed by an agent and a plan library. One component of the plan library representation is a probabilistic model of the choices the observed agent could make when selecting and executing plans from the plan library. A neural network learns this probabilistic model, whereas the rest of the plan library is handcrafted.

In both approaches, a symbolic inference engine makes the goal or plan predictions, not a neural network. Deep learning is involved only as an auxiliary procedure either to scan individual actions [13], or to learn a probabilistic model [4]. In contrast, in the experiments we discuss herein, a neural network makes all the inference.

To the best of our knowledge, Min *et al.* [18] are among the first to use a goal recognition pipeline only made of a neural network. They use feed-forward n-gram models to learn the player's objective from a sequence of his actions in the Crystal Island game. The follow-up method in Min *et al.* [19] uses Long Short-Term Memory (LSTM) networks, better suited to learn patterns in sequences. In both approaches, the features fed to the neural network were engineered instead of merely being raw player's events such as mouse clicks and key presses. While these methods demonstrate favorable results in a specific domain, they do not include a systematic comparison to symbolic ones.

Amado *et al.* [1] more recently introduced a deep learning pipeline to recognize the goal achieved by a player in different simple games (such as 8-puzzle and tower of Hanoi) from raw images, divided into three steps. First, they convert inputs into a latent space (which is a representation of state features) using a previous auto-encoder algorithm [2]. Its properties are built to be reminiscent of a PDDL state representation. Then, an LSTM network utilizes it to perform a regression task, which is making a goal prediction in the latent space. Finally, the decoder reconstructs the goal image from its representation. While this approach does perform well on simple task-planning problems, it may not be applicable in real-life settings. The method indeed tries to extract an approximate domain structure (states representation reminiscent of a PDDL) from temporal changes in observation sequences, and it is unsure whether or not real data can be exploited to frame such rules.

Although some papers started to investigate deep learning for goal recognition, we are not aware of any systematic comparison between an end-to-end deep-learning

pipeline and a symbolic/hybrid approach (in particular, directly and only on raw observations, which is the experiment specifically discussed herein).

2.3 Background

To understand the methodology used for the experiments, we first present some background on deep neural networks and cost-based goal recognition.

2.3.1 The Problem

The goal recognition problem consists in inferring the goal pursued by an actor from an observed sequence of action effects (and sometimes extract the plan pursued by the actor from these, extending the concept to plan recognition) [24]. There is a close link between goals, plans, and intentions. A plan is a sequence of actions achieving a goal, whereas an intention is a commitment to executing a plan. In general, one can infer a goal from a plan and vice-versa. Thus, in the AI literature, plan recognition has come to encompass all problems related to understanding goal-oriented behaviors, whether the focus is on inferring the goal, inferring intention, predicting the plan, or combinations of those three.

The experiments discussed herein deal with inferring the distribution probability of goals by observing action effects. Given a sequence of observations $o_\pi = o_1, \ldots, o_n$, – that may come directly from sensors or followed by relative prior parsing and processing – and a set G of potential goals that the agent might pursue, the problem is to infer a posterior probability distribution across G, $P(G|o_\pi)$, representing the probabilities that the agent might be pursuing a goal given the observations. Note that a goal recognition problem is also a pattern recognition problem, but not vice-versa. That is, not all pattern recognition algorithms harness goal-directed behaviors, let alone, towards inferring the goals underlying goal-directed behaviors.

2.3.2 Deep Learning

It is easy to cast a goal recognition problem as a supervised deep-learning problem. Given a set of sequences of observations $\mathcal{O}$ and a set of potential goals G, let us assume

that there exists a true recognition function f that maps perfectly each $o_\pi \in \mathcal{O}$ to its true goal $g_{o_\pi} \in G$, that is, $f(o_\pi) = g_{o_\pi}$.

While f is unknown (this is what we want to infer), we assume we have access to a training dataset of paired examples (o_π, g_{o_π}) (we know the real goal g_{o_π} for some $o_\pi \in \mathcal{O}$). A supervised learning algorithm will seek to approximate f with a function f' parameterized by some set of parameters θ that minimizes the number of erred predictions in our dataset of examples. In other words, f' minimizes:

$$L = \sum_{n=0}^{N} l(f'(o_\pi^n; \theta), g_{o_\pi^n})$$

where l is a loss function that is 0 when f' predicts accurately, and > 0 otherwise.

A single-layer neural network uses a simple linear transformation of the input using weight and bias parameters followed by a non-linear function in place of f':

$$f'(o_\pi) = \gamma(W o_\pi + b)$$

where W and b are the weight and bias parameters, respectively, and γ is a non-linear function such as sigmoid, hyperbolic tangent (tanh), linear rectifier units (ReLU), or softmax. A (deep) neural network is a composition of several of these transformations, usually with a different set of parameters at each layer [12]. These parameters are trained to minimize the loss function with a gradient descent:

There exist specialized types of networks that process data differently and are more fit for some forms of input and problems. For instance, convolutional neural networks (CNNs) use filters of parameters and the convolution operation to process 2D input, such as images or spatial information. Recurrent neural networks (RNNs) can memorize an internal state and process sequences, such as observed actions, making them better adapted to analyze dynamic behaviors than simple feed-forward architectures are. Long Short-Term Memory networks (LSTM) used by Min *et al.* [19] are types of RNNs that allow for better gradient propagation and thus show better learning results than vanilla RNNs on longer sequences.

2.3.3 Symbolic Cost-Based Goal Recognition

The intuition behind cost-based goal recognition is the *principle of rationality*: people tend to act optimally to the best of their knowledge [3] and motor skills. Thus, one could infer the goal of an observed agent by trying to reason from their point of view, that is, trying to invert his planning process. It does not mean that we need to know his planning process.

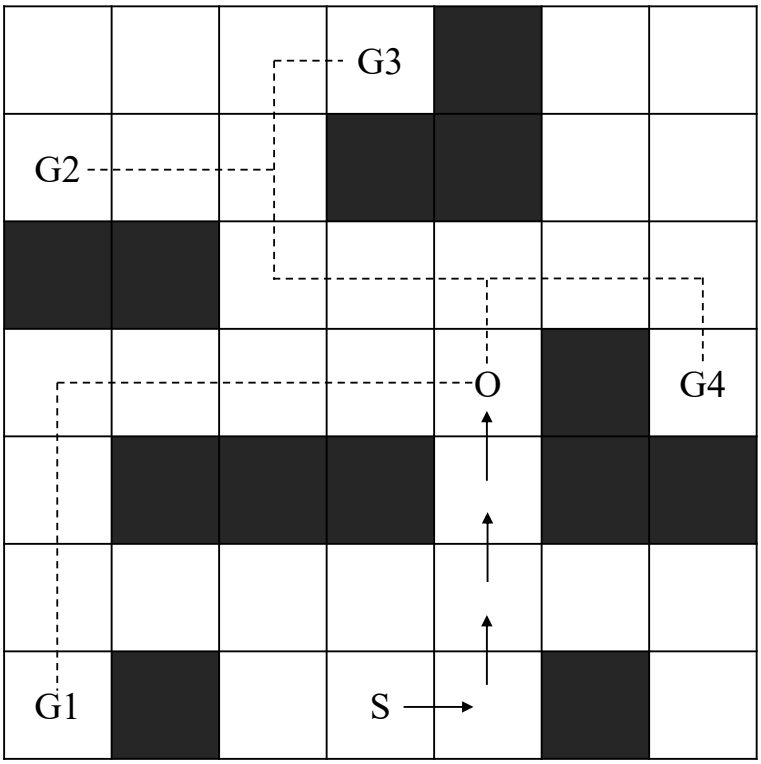

Figure 2.1 – A navigation grid example, where the agent is constrained with obstacles.

As noted by Ramírez and Geffner [22], given a sequence of observations, we could infer the probability that a given goal is the one being pursued by an agent by evaluating if his behavior observed so far is economical and might indeed commit to reaching that goal. To illustrate, consider the map in figure 2.1, representing areas of interest (goals) $G_1, \ldots, G_4$, obstacles, and a sequence of observations of an agent moving around, starting from position S. From the observation so far $o_\pi = o_1 \rightarrow$

$\ldots \rightarrow o_4$, the agent logical goal is unlikely G_1, since we can find a shorter path from its start state to G_1 than the one they are currently taking. Intuitively, we can derive the likelihood of a goal by comparing the cost of an optimal plan consistent with the observations and the cost of an optimal plan not considering the observations. The higher the difference between these two costs is, the less likely the goal is. Formally, Ramírez and Geffner [22] calculate the likelihood of an observation sequence $\mathcal{O}$ to reach a goal g as:

$$P(o_\pi|g) = \frac{e^{-\beta\Delta(s,g,o_\pi)}}{1 + e^{-\beta\Delta(s,g,o_\pi)}}$$

where β is a positive constant determining how optimal we assess the observed agent's behavior to be. Δ is defined to be:

$$\Delta(s, g, o_\pi) = c(s, g, o_\pi) - c(s, g, \neg o_\pi)$$

where $c(s, g, o_\pi)$ is the cost of the optimal plan π_o between s and g that complies with the observations (all observed actions of o_π are embedded monotonically in the plan) and $c(s, g, \neg o_\pi)$ is the cost of the optimal plan $\pi_{\neg o}$ that does not comply with the observations (π does not embed o_π).

From $P(o_\pi|g)$, we can derive the posterior probability of the goal using the Bayes rule: $P(g|o_\pi) = \alpha P(o_\pi|g)P(g)\forall g \in G$, where $P(g)$ is the prior probability (often assumed to be uniform) and α is a normalization factor.

In principle, a planner can be used to compute plan costs [22]. However, calculating a plan, even in the simple case of a deterministic environment under full observability, is NP-Complete [8]. It is not realistic in situations where an agent needs to infer the intention of others quickly. Approximate plan costs, computed by suboptimal planners that run faster than optimal ones, can be used to deduce approximate distribution [21]. They can be helpful in situations where the most important thing is to identify the most likely goals. Nonetheless, even heuristic planners that compute suboptimal plans still take too much time for most real-time applications.

We can avoid some calls to the planners by incorporating heuristic functions directly into the inference process. Vered and Kaminka [30] introduced such heuristics that judge whether a new observation may change the ranking of goals and whether

to prune a goal or not. However, they become useless in more complex problems where the goals cannot be pruned early and do not reduce the number of calls to the planner.

A practical approach to cost-based goal recognition is to compute the plan costs offline. This way, instead of invoking a planner, we have a lookup in a table or a map of plan costs. For navigation problems, where the issue is to predict the destination of an agent moving around, Masters and Sardiña [17] describe an approach for accurately pre-computing plan costs by relaxing Ramírez and Geffner [22]'s algorithm with – practically – no loss in accuracy. It is overall the same, but they compute the cost difference to instead be $\Delta(s, g, n) = c(n, g) - c(s, g)$ where n corresponds to the last seen position of the observed agent. This relaxation not depending on the whole observation sequence avoid computing as many different costs as needed by Ramírez and Geffner [22], making them easier to be stored beforehand. However, it is quite limited in application to the – discrete – navigation domain.

In general, however, there is no well-known method of accurately pre-computing and storing plan costs for all possible combinations of initial and goal states for an arbitrary domain. Sohrabi *et al.* [26] compute the top-k plans for each goal and calculate the goal inference by summing the probability of plans in the set achieving this goal, where the likelihood of a plan does not only depend on its cost but also to what degree it complies to the observations. The problem is that the required number of plans is high (1000) to have results comparable to Ramírez and Geffner [22]'s. Other various recent studies present different ideas to reduce planners' compute time. For instance, E.-Martín *et al.* [9] calculate cost interaction estimates in plan graphs, while Pereira *et al.* [20] use landmarks, with the idea that goals with a higher completion ratio are the likely ones. However, their solutions are less accurate since they are mere approximations of plans generated by an optimal planner.

2.4 Comparison Methodology

To compare cost-based goal recognition to deep learning, we used five synthetic domains often selected to evaluate the performance of a symbolic plan recognizer, as referenced above. Ultimately, we want to examine plan recognizers using real-world

benchmarks. Meanwhile, the synthetic domains can provide some useful insight.

1. Navigation: Predicting the goal destination of an agent navigating a map [16]. The domain consists of 20 maps from StarCraft, provided by MovingAI [4], down-scaled to 64x64 pixels, where the agent can perform actions limited to the first four cardinal directions. We generated the goal recognition problems by placing one initial position and five goals on the maps.
2. Intrusion Detection: Predicting the goals of network hackers with their activities [10]. The observed agent is a user who may perform attacks on ten hosts. There are six possible goals that the hacker might reach by performing nine actions on those servers. Observation sequences are typically between 8 and 14 observations long.
3. Kitchen: Inferring the activity of a cook in a smart home kitchen [34]. The cook can either prepare breakfast, lunch, or dinner (possible goals) [34]. He may manipulate objects, use them, and perform numerous high-level activities. Observation sequences are typically between 3 and 8 actions long.
4. BlocksWorld: Predicting the goal of an agent assembling eight blocks labeled with letters, arranged randomly at the beginning [21]. Achieving a goal consists in ordering blocks into a single tower to spell one of the 21 possible words by the use of 4 actions. Observation sequences are typically between 6 and 10 actions long.
5. Logistics: Predicting package delivery in a transport domain. Six packages must be conveyed between 6 locations in 2 different cities, using one airplane, two airports, and two trucks [21]. There are six possible actions available to achieve ten distinct goals. Observation sequences are typically between 16 and 22 actions long.

The observation data for the four last benchmarks are available at https://github.com/pucrs-automated-planning/goal-plan-recognition-dataset.

For the navigation benchmark, we used four different neural network architectures (see figure 2.2): a fully connected network (FC), an LSTM network, and two convolutional neural networks (CNN). We felt both the LSTM and CNN appropriate for

4. MovingAI Lab: https://movingai.com/

this domain, given that the former usually performs well learning from sequences, whereas the latter is suitable to learn from spatial data (maps in our case).

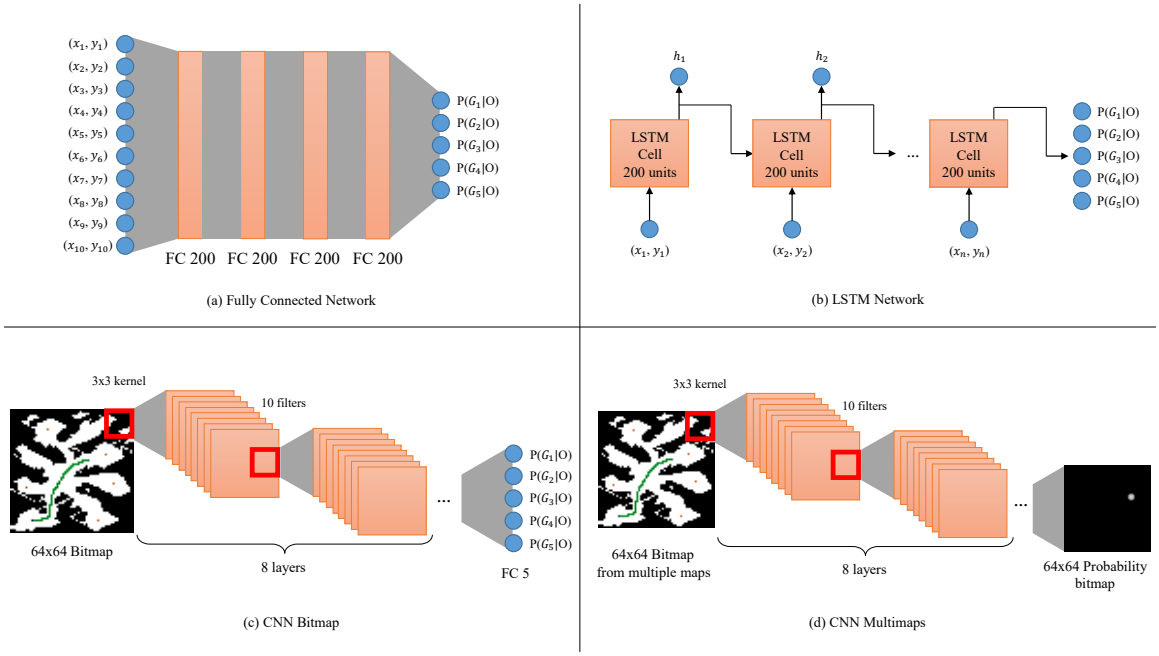

Figure 2.2 – Representation of our architectures for the navigation domain. (x_i, y_i) stands for the coordinates of the agent's location in the grid. (a), (b), and (c) were trained on a single map, while (d) was trained on multiple maps.

We trained the first three networks on problems generated from a single map. We additionally trained a convolutional network (CNNMultimaps) on multiple ones, regardless of their goals, start and obstacle positions, to see if and how it could generalize across multiple navigation domains.

Here is a thorough description of the network architectures:

1. FC: this network contains four dense layers of 200 units and one output layer of 5 units representing the goal probability distribution.
2. LSTM: this network as a single LSTM layer of 200 units and a dense output layer of 5 units.
3. CNN (CNNBitmap): this network has eight convolutional layers of 10 filters of size 3x3, respectively. The resulting features are flattened and passed to a dense layer of 5 units.

4. CNNMultimaps: the first eight layers of this network are the same as in the CNNBitmap, followed by an additional convolution layer of one 3x3 filter instead of a dense layer.

Since we trained and tested the methods FC, LSTM, and CNNBitmap on the same map, with goals identified in advance, it was possible to deduce a probability distribution array of fixed size (five here). However, we could not make this assumption for the general fully convolutional method (CNNMultimaps) trained on multiple, different maps, which instead outputs a probability distribution over the entire grid, representing a spatial belief about the agent's goal, allowing any number of goals and positions in general.

For the four other domains, we used a fully connected network with three dense layers of 256, 32, and 5 units, respectively. We compare it with original Ramírez and Geffner [22]'s method, since there is yet no proven method for pre-computing plan costs – or approximations of them – for these domains without a significant loss in accuracy [9, 20, 30].

Besides the architecture, implementing neural networks involves the choice of specific parameters, activation functions, and optimization algorithm. Given that we want to find a correct goal amongst a set of possible ones and work with probabilistic scores, we quantify the loss with the categorical cross-entropy function and work with the accuracy metric, which is the percentage of correct predictions. A prediction is said to be correct if its highest output probability corresponds to the true goal. In case of ties, we consider a random uniform draw between all the goals having the same top probability. In cost-based goal recognition literature, alternative accuracy metrics are often used, such as metrics using a threshold [20, 26], or simply an accuracy metric where ties are not randomly disambiguated and instead considered as an accurate prediction [22, 9, 26]. However, we find them highly artificial and unfit to evaluations of real-world applications, so we chose to consider the top 1 only, which should account for lower accuracy values. It is also important to note that we apply the same metric to every method.

Hidden layers are activated with the ReLU function, while the output layer is activated with the softmax function. To train the networks, the Adam optimizer [15] is used, with a learning rate of 0.001, $\beta 1$ of 0.9, $\beta 2$ of 0.999 and no decay. To prevent

overfitting, we also used dropout [27] for all layers with a drop chance set to 0.1 or 0.2. Finally, inputs were shuffled uniformly before training.

2.5 Experiments and Results

We present the experiments and discuss their results in this section, including complete details about the training and test datasets. For all domains, the datasets are split 80%-20% for training and test.

2.5.1 Navigation Domain

As mentioned above, we trained four networks for the navigation benchmark. The first three (FC, LSTM, CNNBitmap) were trained for 15 epochs on observations from a single map, with 100 observed paths. We also trained CNNMultimaps on all the available maps for 100 epochs. To mimic suboptimal behavior, we started by generating noisy optimal paths to these goals with a modified A* algorithm, using what we define as an ϵ-over-estimating heuristic:

Definition 2.5.1 *An ϵ-over-estimating heuristic is a function that returns an admissible quantity h' with a chance of $1 - \epsilon$, and $h' + \delta$ otherwise, where $\epsilon \in [0, 1]$ and $\delta > 0$.*

In practice, $\epsilon = 0.2$ and $\delta = 10$.

We truncated the generated paths to measure how our networks could handle early predictions in an online application: both training and test sets consist of partial or complete sequences of observations truncated at the first 25%, 50%, 75% and 100% of the sequence, such that we can evaluate performances for partial as well as complete observability. It is important to note that this notion of partial observability differs from the usual literature definition: in many papers [22, 20, 9, 26], a certain percentage of observations is missing, but across the *whole* sequence. In opposition to that, to mimic real-time predictions, we cut the observation sequences to a given percentage, and drop every following observation. We estimate that this idea of early observability is more realistic as it enables online resolution of goal recognition problems.

We used (x, y) coordinates as input for the FC network and LSTM methods. As paths lengths may differ, we eventually retained a fixed number of positions among the ones available to form inputs of fixed size, padding shorter sequences with zeros. We fed 4-channel bitmaps to both CNNs, where each channel embeds information about either the initial position, the potential goals, the observations, and walkable locations that are neither of the above.

For Masters and Sardiña [16]'s method (labeled M-S), we only considered the last position of the sub-paths. Cost maps were generated using optimal paths returned by the A* algorithm and stored offline. To compute the posterior probabilities, we assumed prior probabilities to be uniform and used a value of 1 for the β parameter.

We compared the accuracy of those four different networks on test sets with M-S. Results are shown in figure 2.3. The Y-axis represents the average accuracy on ten different maps. The X-axis refers to the percentage sampled from total paths in the test set.

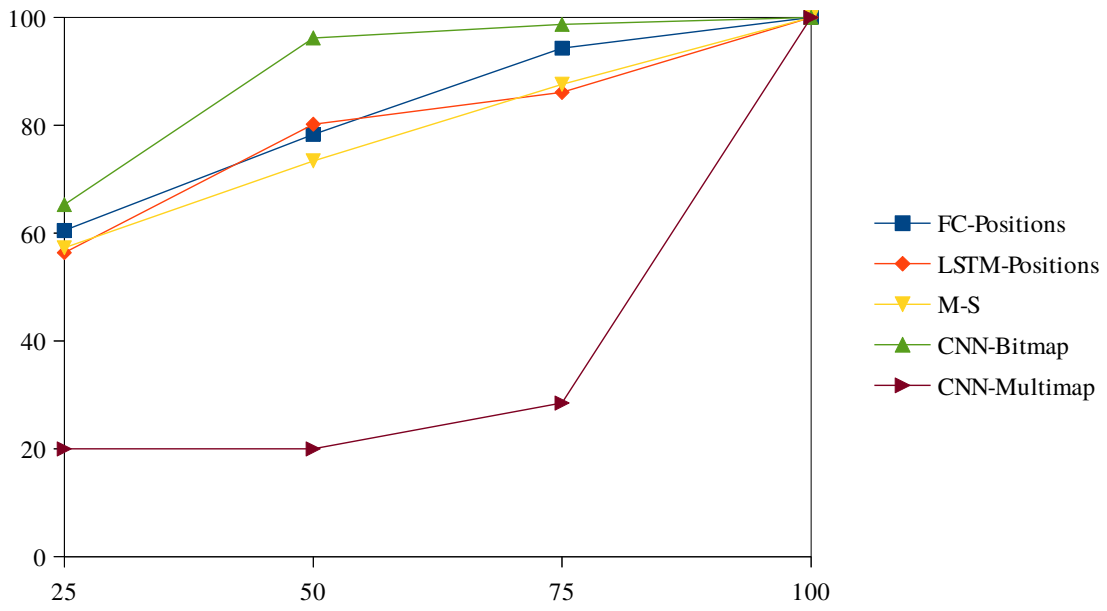

Figure 2.3 – Results of accuracy depending on the percentage retained from the complete observed path, in the navigation domains.

As can be seen, method CNNBitmap ranks first. The reason could be that the

convolution filters of the network help reason about the 2D structure of the grid and the observed path, as expected. FC and LSTM methods perform well too, but it seems that learning from coordinates is more complicated, or more imprecise, that learning directly from bitmaps in such a navigation domain.

Surprisingly, M-S was outperformed at least by CNNBitmap and FC. The reason might be that generated A* tracks stayed somehow deterministic despite the noisy behavior, and thus, even in the case where multiple optimal paths to a goal exist, similar routes were always chosen for that goal. The neural networks thus quickly learned to fit these specific paths, even though earlier subsets could go to either goal. This bias in the data incorporated by the generation process could be problematic, but we argue otherwise. In real-world applications involving human agents, people usually take the same road even when multiple ones that are as good – or even better – exist. Data is therefore not uniformly distributed between every candidate road. The capacity of neural networks to learn this bias and adjust for particular contexts and individuals is one of the properties that makes them appropriate for goal recognition in real-life applications. Additionally, in the case of cost-based algorithms, even though all available data is used to compute costs, the final prediction is only achieved based on them, which represents a gradual loss of information.

The convolutional network trained and tested on all maps (CNNMultimaps) shows relatively incorrect early predictions (20% accuracy for five goals is just a random prediction), proving there is still room for improvement to generalize to multiple maps. Nonetheless, the method can already create a link between a complete path and a goal (that is, learning but not predicting), and we may significantly improve its results using specialized architectures, such as value iteration network [29] and visual relational reasoning [32]. We are currently working on improving its results.

Computing plan costs takes time, even offline. The results suggest that training neural networks, even if computationally complex, may be advantageous in this regard thanks to the trivially parallelizable nature of its operations and the computation power of modern hardware. However, a computation time comparison does not enlighten new advantages for this kind of context. Table 2.1 gives a summary of offline and online computation times. The LSTM networks have longer training times but may generalize better to longer sequences of observations with bigger sliding

windows (since we fixed the maximum number of observations input to 10 and thus do not benefit sufficiently from LSTM's training power over sequences). The CNN trained on multiple maps takes a long time to train but could have the potential to generalize to every navigation problem so that it would require no additional training for unseen configurations. Symbolic approaches have no need for training nor dataset, but knowledge about the domain is needed to handcraft the model, and costs must be generated for every new map, whether it is offline or online (during prediction).

	T	P
FC	10 s	10μs
LSTM	30 s	4 ms
CNNBitmap	10 s	4 ms
CNNMultimaps	20 min	4 ms
R-G	0	1 s
M-S	7 s	10μs

Table 2.1 – Comparison of rough average computation times of the evaluated approaches on the navigation domain. T is the offline computation time, while P is the online prediction time.

2.5.2 Other Domains

The navigation benchmark deals with path-planning problems requiring much less knowledge than the other four domains. Those last benchmarks correspond to task-planning problems, involving constraints that differ from those in the navigation benchmark, thus requiring different kinds of domain representations (represented using the Planning Domain Definition Language (PDDL) as in Ramírez and Geffner [22]).

We trained a fully connected network during 15 epochs, with 1000 to 3000 examples depending on each domain. We also trained an LSTM on these examples, but it ended up taking more time without providing significant result improvements.

A training example in the datasets is a sequence of observations from PDDL files. Each observation in the sequence is one action type plus its arguments, both transformed into a one-hot vector. The neural network receives the complete sequence

of transformed observations. To match a fixed input size, sequences shorter than the maximum size are padded with zeros and shifted $maxSize - size + 1$ times (for instance, if one observation is AB and the maximum size is 4, 3 new observations will be created: $AB00$, $0AB0$, $00AB$), hence generating new training data.

In the case of Ramírez and Geffner [22]'s method, labeled R-G, the costs were generated online, as first implemented by the authors, from optimal plans found by the HSP planner. The β parameter value was one, and the prior probabilities of the goals were presumed to be uniform.

Results in figure 2.4 show the accuracy for both methods. The fully connected network outperforms the R-G approach almost every time. We provide a similar explanation for these results: generated sequences tend to be biased for each goal, and the network learned it. In addition to producing higher prediction rates, networks are also quicker: on such problems, the training part takes approximately one minute to infer reusable weights, and a prediction requires approximately 1ms. The R-G approach does not require training nor offline computation, but provides a prediction in minutes, sometimes hours, which is very long and cannot run for real-time decision making. A suboptimal planner might reduce computation times, but we can reasonably assume that it would remain above several minutes or so for each goal prediction.

2.6 Conclusion

Although still preliminary, these results suggest that deep learning outperforms symbolic inverse planning, at least in the five domains considered. We plan to pursue this experimentation in real-world settings where we can gather data, including video games. We also plan to try other deep neural networks [12], symbolic methods, multi-agent configurations, sensor limitations (partial observability vs. full observability), attitudes between the observed agent and the observer (cooperative, adversarial, neutral) and different domains of application.

In some applications, the plan recognizer needs to explain the rationale of its inferences. To do so, extracting a meaningful explanation from a neural network remains a challenge. In contrast, the representation of symbolic plan recognizers

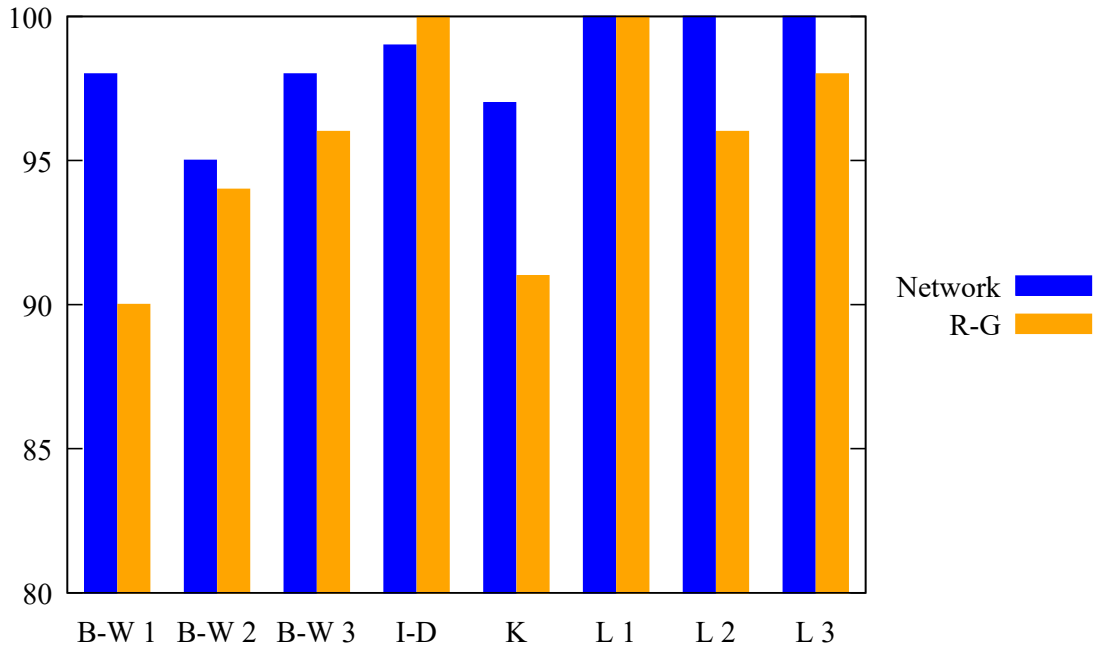

Figure 2.4 – Results of accuracy for the task-planning domains (B-W, I-D, K, and L stand for Blocks World, Intrusion Detection, Kitchen and Logistics respectively).

directly answers to this question, except that, as we have argued, those approaches are difficult to ground in real-world environments. It suggests that the exploration of hybrid approaches, such as those discussed in the related section, remains worth pursuing.

2.7 Acknowledgements

The Natural Sciences and Engineering Research Council (NSERC) of Canada and the *Fonds de recherche du Québec – Nature et technologies* (FRQNT) supported the work with grants, and Compute Canada provided computing resources. The NVIDIA Corporation donated the Quadro P6000 used for this research. We are also thankful to Julien Filion, Simon Chamberland, and anonymous reviewers for their insightful feedback that helped improve the paper.

References

[1] L. Amado, J. P. Aires, R. F. Pereira, M. C. Magnaguagno, R. Granada, F. Meneguzzi, "LSTM-Based Goal Recognition in Latent Space," *CoRR*, vol. abs/1808.05249, 2018. [Online]. Available: http://arxiv.org/abs/1808.05249

[2] M. Asai A. Fukunaga, "Classical Planning in Deep Latent Space: Bridging the Subsymbolic-Symbolic Boundary," in *AAAI 2018*, 2018. [Online]. Available: https://aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16302

[3] C. L. Baker, R. Saxe, J. B. Tenenbaum, "Action understanding as inverse planning," *Cognition 2009*, vol. 113, no. 3, pp. 329–349, 2009.

[4] F. Bisson, H. Larochelle, F. Kabanza, "Using a Recursive Neural Network to Learn an Agent's Decision Model for Plan Recognition," in *IJCAI 2015*, 2015, pp. 918–924.

[5] H. H. Bui, S. Venkatesh, G. A. W. West, "Policy Recognition in the Abstract Hidden Markov Model," *JAIR*, vol. 17, pp. 451–499, 2002.

[6] E. Charniak R. P. Goldman, "A Bayesian Model of Plan Recognition," *Artif. Intell.*, vol. 64, no. 1, pp. 53–79, 1993. [Online]. Available: https://doi.org/10.1016/0004-3702(93)90060-O

[7] C. Chen, X. Zhang, S. Ju, C. Fu, C. Tang, J. Zhou, X. Li, "AntProphet: an Intention Mining System behind Alipay's Intelligent Customer Service Bot," in *IJCAI 2019*, 08 2019, pp. 6497–6499.

[8] G. F. Cooper, "The Computational Complexity of Probabilistic Inference Using Bayesian Belief Networks (Research Note)," *Artificial Intelligence*, vol. 42, no. 2-3, pp. 393–405, March 1990. [Online]. Available: http://dx.doi.org/10.1016/0004-3702(90)90060-D

[9] Y. E.-Martín, M. D. R.-Moreno, D. E. Smith, "A Fast Goal Recognition Technique Based on Interaction Estimates," in *IJCAI 2015*, 2015, pp. 761–768.

REFERENCES

[10] C. W. Geib R. P. Goldman, "Requirements for Plan Recognition in Network Security Systems," in *International Symposium on Recent Advances in Intrusion Detection 2002*, 2002.

[11] C. W. Geib R. P. Goldman, "A probabilistic plan recognition algorithm based on plan tree grammars," *Artificial Intelligence*, vol. 173, no. 11, pp. 1101–1132, 2009.

[12] I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning*. MIT Press, 2016.

[13] R. Granada, R. Pereira, J. Monteiro, R. Barros, D. Ruiz, F. Meneguzzi, "Hybrid Activity and Plan Recognition for Video Streams," in *AAAI 2017*, 2017.

[14] J. Hou, X. Wu, J. Chen, J. Luo, Y. Jia, "Unsupervised Deep Learning of Mid-Level Video Representation for Action Recognition," in *AAAI 2018*, 2018, pp. 6910–6917.

[15] D. P. Kingma J. Ba, "Adam: A Method for Stochastic Optimization," *CoRR*, vol. abs/1412.6980, 2014.

[16] P. Masters S. Sardiña, "Cost-Based Goal Recognition for Path-Planning," in *AAMAS 2017*, 2017, pp. 750–758.

[17] P. Masters S. Sardiña, "Cost-Based Goal Recognition in Navigational Domains," *JAIR*, vol. 64, pp. 197–242, 2019. [Online]. Available: https://doi.org/10.1613/jair.1.11343

[18] W. Min, E. Ha, J. P. Rowe, B. W. Mott, J. C. Lester, "Deep Learning-Based Goal Recognition in Open-Ended Digital Games," in *AIIDE 2014*, 2014.

[19] W. Min, B. W. Mott, J. P. Rowe, B. Liu, J. C. Lester, "Player Goal Recognition in Open-World Digital Games with Long Short-Term Memory Networks," in *IJCAI 2016*, 2016, pp. 2590–2596.

[20] R. F. Pereira, N. Oren, F. Meneguzzi, "Landmark-Based Heuristics for Goal Recognition," in *AAAI 2017*, 2017, pp. 3622–3628.

[21] M. Ramírez H. Geffner, "Plan Recognition as Planning," in *IJCAI 2009*, 2009, pp. 1778–1783.

[22] M. Ramírez H. Geffner, "Probabilistic Plan Recognition Using Off-the-Shelf Classical Planners," in *AAAI 2010*, 2010.

[23] A. Sadilek H. A. Kautz, "Recognizing Multi-Agent Activities from GPS Data," in *AAAI 2010*, 2010. [Online]. Available: http://www.aaai.org/ocs/index.php/AAAI/AAAI10/paper/view/1603

[24] C. F. Schmidt, N. S. Sridharan, J. L. Goodson, "The Plan Recognition Problem: An Intersection of Psychology and Artificial Intelligence," *Artificial Intelligence*, vol. 11, no. 1-2, pp. 45–83, 1978.

[25] K. Simonyan A. Zisserman, "Two-Stream Convolutional Networks for Action Recognition in Videos," in *NIPS 2014* Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, K. Q. Weinberger, editors, 2014, pp. 568–576. [Online]. Available: http://papers.nips.cc/paper/5353-two-stream-convolutional-networks-for-action-recognition-in-videos.pdf

[26] S. Sohrabi, A. V. Riabov, O. Udrea, "Plan Recognition As Planning Revisited," in *IJCAI 2016*, ser. IJCAI'16. AAAI Press, 2016, pp. 3258–3264. [Online]. Available: http://dl.acm.org/citation.cfm?id=3061053.3061077

[27] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[28] G. Sukthankar, C. Geib, H. H. Bui, D. Pynadath, R. P. Goldman, *Plan, Activity, and Intent Recognition: Theory and Practice*, 1st edition. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2014.

[29] A. Tamar, Y. Wu, G. Thomas, S. Levine, P. Abbeel, "Value Iteration Networks," in *IJCAI 2017*, 2017, pp. 4949–4953.

[30] M. Vered G. A. Kaminka, "Heuristic Online Goal Recognition in Continuous Domains," in *IJCAI 2017*, 2017, pp. 4447–4454.

REFERENCES

[31] B. Volz, K. Behrendt, H. Mielenz, I. Gilitschenski, R. Siegwart, J. Nieto, "A data-driven approach for pedestrian intention estimation," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, Nov 2016, pp. 2607–2612.

[32] N. Watters, A. Tacchetti, T. Weber, R. Pascanu, P. Battaglia, D. Zoran, "Visual Interaction Networks," *CoRR*, vol. abs/1706.01433, 2017.

[33] T. Wen, Y. Miao, P. Blunsom, S. J. Young, "Latent Intention Dialogue Models," in *ICML 2017*, 2017, pp. 3732–3741. [Online]. Available: http://proceedings.mlr.press/v70/wen17a.html

[34] J. Wu, A. Osuntogun, T. Choudhury, M. Philipose, J. M. Rehg, "A Scalable Approach to Activity Recognition based on Object Use," in *ICCV 2007*, 2007.

[35] S. Yan, Y. Teng, J. S. Smith, B. Zhang, "Driver behavior recognition based on deep convolutional neural networks," in *ICNC-FSKD 2016*, 2016, pp. 636–641.

Chapitre 3

Une méthode de transfert d'apprentissage utilisant des caractéristiques inter-domaines pour la reconnaissance de but

Les citations de ce chapitre redirigent vers les références à la page 61.

Résumé

En considérant les résultats de l'article précédent, il est à présent pertinent de se pencher sur la question de la généralisation. En effet, les réseaux de neurones se montrent performants dans le cadre où ils ont été entraînés, mais sont inefficaces dans des situations jamais vues auparavant.

L'intuition derrière cet article est fondée sur le fonctionnement général du raisonnement humain : l'existence d'une base de connaissances commune à des contextes similaires. Par exemple, un individu ayant appris à lire et écrire manuscritement n'aura aucune difficulté à utiliser les caractères électroniques d'un ordinateur. De la même manière, en apprenant à jouer à un nouveau jeu vidéo pour

la première fois, un humain comprendra directement qu'une porte fermée s'ouvrira avec un déclencheur (clé, levier, ...), puisqu'il en est ainsi dans la vraie vie.

Dès lors, nous proposons une méthode de transfert d'apprentissage se basant sur le principe de *few-shot learning* (apprentissage avec peu d'exemples), à partir d'un réseau pré-entraîné sur un domaine similaire. Nous exploitons les données en les projetant vers un domaine intermédiaire de structure spatiale et utilisons un *CNN* pour exploiter la proximité des points dans l'espace de deux dimensions. Une fois le réseau entraîné sur un domaine de base, nous l'adaptons avec seulement quelques exemples à un domaine similaire, en constatant que ses premières couches ne considèrent que des caractéristiques communes à tous les domaines (formes, motifs, contours, ...).

Nos résultats mettent en valeur la réussite de cette technique, toujours dans le même domaine de navigation synthétique, qui arrive presque au même niveau qu'un réseau complètement ré-entraîné.

Commentaires

Cet article a été publié sur arXiv [1]. Ce projet a été entièrement mené par Thibault Duhamel, supervisé par Froduald Kabanza, avec les retours et les suggestions de Mariane Maynard.

1. https://arxiv.org/abs/1911.10134

A Transfer Learning Method for Goal Recognition Exploiting Cross-Domain Spatial Features

Thibault Duhamel, Mariane Maynard, Froduald Kabanza
Département d'informatique
Université de Sherbrooke
Sherbrooke, Québec (Canada) J1K 2R1
thibault.duhamel@usherbrooke.ca,
mariane.maynard@usherbrooke.ca,
froduald.kabanza@usherbrooke.ca

Abstract

The ability to infer the intentions of others, predict their goals, and deduce their plans are critical features for intelligent agents. For a long time, several approaches investigated the use of symbolic representations and inferences with limited success, principally because it is difficult to capture the cognitive knowledge behind human decisions explicitly. The trend, nowadays, is increasingly focusing on learning to infer intentions directly from data, using deep learning in particular. We are now observing interesting applications of intent classification in natural language processing, visual activity recognition, and emerging approaches in other domains. This paper discusses a novel approach combining few-shot and transfer learning with cross-domain features, to learn to infer the intent of an agent navigating in physical environments, executing arbitrary long sequences of actions to achieve their goals. Experiments in synthetic environments demonstrate improved performance in terms of learning from few samples and generalizing to unseen configurations, compared to a deep-learning baseline approach.

3.1 Introduction

Goal recognition is a critical feature of intelligent agents, as it allows them to anticipate future behaviors that have not yet been observed and integrate *a priori* knowledge to make informed decisions. It is a fundamental cognitive ability lying

at the heart of social interactions, often unconsciously, unlocking the possibility to understand beyond explicit communication. While humans intuitively manage to implicitly recognize and predict a course of action of others by observing them, granting machines such a strong ability remains a challenge.

This problem conveys several dimensions of complexity. Short-term action recognition, for instance, focuses on identifying activities over a short horizon using low-level sensors [17, 38], as opposed to long-term goal recognition that aims to predict sequences of actions over a longer horizon [33]. Behaviors can be fully or partially observable [15] and may involve multiple agents, cooperating or competing with each other [8].

In this paper, a single observer tries to infer the goal of a single agent evolving in a neutral environment, with a behavior either fully or partially observable. Traditional symbolic approaches to this type of goal recognition problem require handcrafted knowledge, engineered by experts, conveying the space of potential behaviors of the observed agent [27, 20, 25, 30, 35, 36]. Unfortunately, it has proven difficult to express such knowledge in practice since a part of the human decision making is unconscious, hence impossible to model explicitly.

Using deep learning to learn from data is an attractive alternative approach, not impeded by the limitations of handcrafted models. Deep learning is used, for instance, to classify intents of utterances in natural language processing [7, 39] or activities in video analysis [19]. The sequentiality is a common feature of these two types of applications, as natural language consists of sequences of words, whereas activities in video analysis are short sequences of low-level actions. In contrast, there have been so far fewer research efforts to apply deep learning to infer goals behind long sequences of actions. The rare existing approaches use conventional deep learning architectures such as convolutional or long-short term memory networks, with the difficulty of being able to generalize across domains [22, 1]. For example, while it is possible to predict the goals of others walking around in one particular scenario, the learned model does not apply to a completely different one, let alone a new one with fewer samples.

Following a similar inquiry to these previous approaches, this paper aims to demonstrate how it is possible to combine transfer learning and few-shot learning

to infer the goal of agents engaged in long navigation behaviors in a physical environment. Transfer learning consists in reusing a model optimized for a specific domain as a starting point and somehow adapt it for another one. While quite well understood for many applications, including NLP [13], image analysis [11], and others [34], this technique has, to the best of our knowledge, never been used for inferring the goal of agents engaged in long-term behaviors such as in the navigation domain. Few-shot learning, on the other hand, consists of optimizing a learner with a significantly reduced amount of examples [28]. As far as we know, no paper applied this technique for long-term goal recognition.

Humans are efficient at learning to perform a variety of tasks in different domains, using a shared base of knowledge that is transferable (with perhaps a few examples). For instance, in video games, when a player faces a door, he intuitively searches for a key that might open it, leveraging related information gathered in his everyday experiences.

Likewise, we here use few-shot transfer learning to train a deep neural network in such a way that it would learn cross-domain intent patterns that are quickly adaptable to other scenarios in a navigation domain, with fewer training samples compared to baseline networks. To the best of our knowledge, no previous deep learning approach to goal recognition in the navigation domain (let alone any other domain conveying agents driven by long-term planning) has demonstrated an ability to generalize across different scenarios from a few examples.

To test this idea, we use a synthetic grid-world environment similar to previous approaches so far [21]. However, instead of using a symbolic map as input for the goal recognizer, our approach utilizes raw bitmaps of higher resolution, similar to images or video-game deep learning applications [37]. By doing so, the input framework in our approach is closer to real-world settings while being comparable to previous approaches.

We organize the rest of the paper as follows: first, we discuss the most relevant approaches while providing background concepts. Then, we describe our approach, followed by the experiments, along with a discussion.

3.2 Related Work

A large body of recent research for goal inference still uses symbolic knowledge representations and inferences. While we apply deep neural networks, it is useful to contrast both paradigms.

3.2.1 Symbolic Goal Recognition

Symbolic approaches to goal recognition traditionally cast inference processes as abductive reasoning, from observations to goals or plans, using some causal reasoning framework [33]. A goal recognizer thus has two main components: (1) domain knowledge in some formal reasoning formalism, characterizing the potential behaviors of the observed agent; (2) an inference algorithm for reasoning about the knowledge, to infer goals or plans from observations.

The amount of knowledge required by these different approaches may vary. Most approaches request, in one way or another, knowledge about both the primitive actions of the observed agent and the rules governing its behaviors, also called plan libraries. Approaches based on Bayesian networks [6], hidden Markov models [5], Markov logic [31], hierarchical task networks (HTN) [3], probabilistic grammars (which are in essence equivalent to HTNs augmented with a probabilistic model) [9, 14] can be grouped into that category. The so-called cost-based approaches or inverse-planning approaches only need a model of primitive actions of the observed agent, but not the plan library [27, 21, 25, 30, 35, 36].

A common issue with all these approaches is the ability to represent the domain knowledge symbolically. Human beings, including experts, have difficulties accessing the unconscious mechanisms that participate in their decision making processes. It is a tremendous challenge for many domains – those involving image recognition in particular – to specify a knowledge-based model that can support goal inference in practice.

3.2.2 Deep Learning of Models for Symbolic Goal Inference

A natural thought about trying to overcome the knowledge-engineering challenge is to learn models for goal recognition. Bisson *et al.* [4] experimented with the idea of using recursive neural networks to learn the probabilistic model underlying an HTN plan library used by a symbolic probabilistic goal recognizer. Granada *et al.* [10] created, in a kitchen environment, a hybrid technique using a deep neural network to identify independent actions from sensors and the SBR algorithm (Symbolic Behavior Recognition) to recognize the goal achieved from the sequence of observations, with a plan library. Pereira *et al.* [26] proposed to use a neural network to learn a nominal model (states and transition rules) of the environment and perform goal recognition with a planner on this model.

3.2.3 End-to-End Deep-Learning for Goal Recognition

This paper is interested in an end-to-end deep learning pipeline for goal recognition. Such an approach appears increasingly attractive, in the wake of recent breakthroughs solving complex games like Go [29] and real-time strategy games [37].

Various applications of video analysis show that deep learning is making significant inroads in recognizing short activities performed by people [19, 38, 17]. Nevertheless, the discussion here focuses on applications for long-term behaviors.

Long Short Term Memory networks (LSTM) have been used to recognize the goal of quite long-term behaviors by a player in the CRYSTAL ISLAND open-world game Min *et al.* [22, 23]. Using data collected from in-game interactions, an LSTM was trained to predict the player's goal from his sequence of interactions, with reliable performance.

Amado *et al.* [1] introduced a pipeline to recognize the goal achieved by a player in different simple games (such as 8-puzzle and tower of Hanoi) from constructed images of the game state, divided into three steps. First, they convert inputs into a latent space (which is a representation of state features) using a dense auto-encoder network previously introduced in Asai and Fukunaga [2]. Then, an LSTM network utilizes this representation to perform a regression task, consisting of constructing a goal prediction in the latent space. Finally, a decoder network reconstructs the image

of the goal from its latent representation.

While these learning architectures demonstrate impressive capabilities to perform goal recognition, they are unable to generalize to previously unseen configurations, as we shall illustrate with examples in the navigation domain. It is an essential point because an environment might change or slightly evolve (game updates, for instance). Another limitation, closely related, lies in the amount of data required to train a neural network.

3.3 Proposed Method

The fundamental motivation behind transfer learning is to reduce the effort needed to label new data for scenarios never seen beforehand, which is crucial in real-life applications. We believe similar domains must share identical features, as it is the case in image classification [18]: no matter what classes are to be recognized, there are always patterns like edges, lines, and other shapes involved in the learning process. From this point forward, it seems counterproductive to learn those features again for different targets.

Our approach for a deep learning framework with improved generalization capability and reduced training examples consists of combining transfer learning and few-shot learning. With a data representation tweak allowing to convey temporal information into a trajectory trail, we show that navigation goals can be inferred by a CNN alone, without an LSTM, along with a generalization ability to different navigation maps using reduced training samples.

3.3.1 Deep Learning Architecture

A convolutional neural network (CNN) is a specific deep learning architecture designed to exploit spatial proximity using the convolution operation. Filters of parameters are shared by translating a kernel across the dimensions of the input, especially advantageous with images or, in our case, 2D grids:

$$h_{x',y'} = \sigma(W * i) = \sigma(\sum_{x,y=0}^{M,N} W_{x,y} i_{x'-x,y'-y})$$

where h is the output matrix, σ is the activation function, W is the kernel of size (N, M), $*$ is the convolution operation and i is an input window.

Our stacked data representation motivates our choice to use a CNN for goal recognition since the property of local connectedness is a convenient component to identify links between adjacent pixels, whether they are consecutive observations, walls, starts, or goals.

3.3.2 Data Representation

Projecting data to a subs-space (or latent space) to reduce the dimension by abstracting over irrelevant information that impairs the learning process is a technique often used in deep learning [24, 41].

In our approach, we represent a sequence of observations as a stacked spatial trail on a bitmap, thus projecting the temporal dimension on a 2D space. As far as we know, Liu *et al.* [19], Yan *et al.* [40] initially introduced this representation in short-term activity recognition, converting the time axis to a third spatial dimension for a 3D convolutional network, such that to exploit the closeness between two consecutive states for a given pixel.

In our navigation domain, we suspect there is a link between two successive events and two points of short distance, specifically in the navigation domain. In other words, the spatial closeness is equivalent to temporal proximity in the studied context, or at least sufficient to lose no equivalent information. Thus, in a stacked trajectory trail, the temporal information is conveyed by the trail, suggesting that a CNN, instead of a CNN combined with an LSTM, would be enough to learn the features relevant for goal recognition, saving us computation and memory resources.

Given a sequence of observed positions $O = \{o_1 = (x_1, y_1), ..., o_n = (x_n, y_n)\}$, a list of obstacle coordinates C, a start position S and a list of 10 possible goals G, we build a 5-channels bitmap $(B_{i,j})_{i,j\in[1,N]}$ where:

— $B_{i,j} = (1, 0, 0, 0, 0)$ *if* $(i, j) \in C$
— $B_{i,j} = (0, 1, 0, 0, 0)$ *if* $(i, j) \in O$
— $B_{i,j} = (0, 0, 1, 0, 0)$ *if* $(i, j) = S$
— $B_{i,j} = (0, 0, 0, 1, 0)$ *if* $(i, j) \in G$

— $B_{i,j} = (0, 0, 0, 0, 1)$ *otherwise (navigable tiles)*

The intuition that an image of a trajectory trail fed to a CNN might be enough to learn to infer the goal destinations of an observed agent, without the additional use of an LSTM, appears logical from a human cognition standpoint. When people usually depict a trajectory on a map, they intuitively draw lines representing a flattened version of their temporal reasoning, implicitly considering the time steps. That way, the goal recognition problem of an agent navigating in a map is transformed into recognizing motion patterns, which, as we demonstrate later, a convolutional network can learn to extract.

3.3.3 Few-Shot Transfer Learning

The adaptation process commences with a basic gradient optimization of a convolutional network on a single map, with a fixed configuration of start, obstacles, and goals, with several examples provided for every possible goal. We refer to this data as the base training set and base testing set, of respective sizes 16000 and 12800. The network, hence trained during five epochs, is called the *base network* and will be the one to adapt in the subsequent steps.

Our critical hypothesis is that there could be patterns (what we name cross-domain features) identified by the first layers of the base network, that are not involved in the mere goal recognition process. On the contrary, similarly to image classification, they would only recognize key edges, lines, shapes, or points somehow derived from the raw bitmap input. These features, of course, would not be specific to just one scenario and should be preserved when adapting the network to a new one.

From this assumption, we aim to quickly adapt the base network with only a few shots for an unseen configuration of obstacles, start, and goals (named transfer training set and transfer testing set). The same configuration is used in the transfer training set and the transfer testing set but is different from the one used in the base sets. The transfer training set contains n shots, where a shot consists of one example per goal and observability (25%, 50%, 75%, and 100%, see next section). There are thus $4n|G|$ different examples in the transfer training set. The transfer testing set also contains 12800 examples to evaluate transfer performances. To adapt the base

network, we freeze a certain amount of the first layers, which means only the last ones will be affected by a new gradient optimization using the transfer training set for three epochs only.

In the experiment section, we evaluate the performance of the adapted network depending on three hyperparameters that we tune: the number of locked layers, the number of shots provided, and the transfer learning rate. We work with the accuracy metric, which is the ratio of correct predictions over the total number of predictions performed. A prediction is correct when its highest assigned probability score corresponds to the real goal. In case of ties, a random draw applies.

3.4 Experiments

We conducted the experiments on the navigation domain, one of the benchmarks currently used by state-of-the-art goal recognition algorithms [21]. The problem consists in predicting the destination of an agent moving on a map, given its trajectory so far (observations). We downloaded 30 StarCraft maps from the MovingAI website [32] [2] and downscaled them to 512x512 pixels, in which the agent can move up, down, left or right to reach one goal amongst a set of 10 possible ones, randomly sampled. As mentioned above, we aim to increase the input resolution so that our examples become more realistic than toy ones.

Though still synthetic, we aim to tighten the frontier between generated and real-world data by using no handcrafted expert knowledge at all, providing as input for our networks just the pixels of the computed bitmaps (see figure 3.1), made of 5 channels to represent either a navigable tile, an obstacle, an observation, a start or a goal. Moreover, we introduced noise in the agent's behavior by generating its path with a modified version of A* with a chance to drop an optimal step and pick a non-optimal one, to mimic a human-controlled route, using what we define as an ϵ-over-estimating heuristic:

Definition 3.4.1 *An ϵ-over-estimating heuristic is a function that returns an admissible quantity h' with a chance of $1 - \epsilon$, and $h' + \delta$ otherwise, where $\epsilon \in [0, 1]$ and*

2. MovingAI Lab: https://movingai.com/

$\delta > 0$.

In practice, $\epsilon = 0.2$ and $\delta = 10$.

We begin by verifying the key hypothesis that lies behind our transfer learning idea for goal recognition with a qualitative analysis. To validate the existence of a common base of knowledge, we initially trained a CNN network (figure 3.2) on one configuration of the navigation domain and displayed its activation layers, from input to output. This CNN is a succession of 7 convolutional layers with 16 filters of 3x3 kernels interspersed with ReLU activations, followed by a final dense layer of 10 units with a softmax activation. It was optimized using Adam [16] with a learning rate of 0.01, $\beta_1 = 0.9$ and $\beta_2 = 0.999$, no decay, minimizing the cross-entropy loss.

The images obtained from this visualization process (see figures 3.3 and 3.4) suggest that the first layers handle feature extraction (such as observations, walkable areas, walls and edges between them), while the last ones perform the goal recognition task, at least visually. It is intriguing to observe that some filters do not even consider the trail of observations and only focus on the map configuration. In image classification, the same phenomenon reveals that the first layers often recognize edges, curves, and shapes. The pipeline thus progressively increases from low to high-level processing. From this observation, we plan to freeze layers that are not involved in the goal recognition process and re-train those who are.

We hence experimented with several transfer configurations and compared them with a network fully trained and tested on a single domain as a baseline. To begin with, we built a CNN slightly different from the one described above and trained it on a single map configuration. The purpose was to minimize dependencies between layers so that they would be easier to adapt. To do so, we replaced each convolutional layer by a block made of:

— a convolutional layer (same parameters, with a He-Uniform initialization [12])
— a batch normalization for faster optimization and stability. It furthermore helps to reduce the impact of changing the input configuration.
— a ReLU activation
— a dropout chance of 0.1, to increase layer independence

We trained this base network during five epochs of 16000 examples, with paths truncated at 25%, 50%, 75%, and 100% of their full length (see figure 3.1). In

this paper, we do not consider full/partial observability as it is common in literature. Usually, an observability level of x% denotes that x% of the path is observed and taken into account. It means there exists a chance that, for instance, an observability level of 1% only retains the last positions, including the goal. Moreover, it is unadapted to online predictions because the full path is required beforehand. Here, we retain the first x% of the path and study the convergence of our method, comparing online predictions at every step.

The classification target is simply a one-hot vector of size ten, and each output unit of the network is a probability score for each goal, measuring the certainty of its predictions.

We then duplicated this network, preserving its trained weights, and locked a certain number of layers. The remaining set of free weights was adapted to a new configuration, never seen before (different map, starting point, and goals locations). To do so, we introduce some transfer learning hyperparameters controlling the optimization process:

— the number of shots required to adapt the network
— the number of layers that are frozen
— the transfer learning rate

The model was cross-validated by being adapted to 5 different new maps, with 3200 validation examples per map and per combination of hyperparameters. The mean accuracy for each hyperparameter value is studied below, according to the convergence metric just mentioned.

3.4.1 Frozen Layers

The number of frozen layers may impact both the transfer learning quality and the adaptation duration. In this section, we set the number of shots to 5, along with a learning rate of 0.01. We selected those values after manually testing some configurations. Results are shown in figure 3.5.

There appears to be an optimal ratio to discover between the total number of layers and the number of layers to lock. When there are too many free layers (0 or 1 locked layer), the network performs poorly: since there are only a few shots available

to train the free layers, a higher amount of weights should be far more challenging to optimize. Moreover, we assumed that the first layers were capable of handling features, and this experiment demonstrates that re-training those counteracts the adaptation of the entire set of weights.

The opposite is also inconvenient, as locking too many layers (5 or 6) hinders the tuning of the last goal recognition layers.

3.4.2 Number of Shots

The number of shots required to adapt a neural network is crucial in several contexts where critical classes are unbalanced. It is not the case here, but we may think of extreme cases (such as rare diseases and bomb attacks) where it is impossible to gather enough data to train a network from scratch. The human learning process is capable of fast convergence with just one example of a previously unseen entity, and it is a crucial feature we should grant to neural networks.

In this section, we locked five layers in the base network (which means that two are free) and set the transfer learning rate to 0.01. The maximum number of shots we reached was ten since we noticed no improvement after this value. Figure 3.6 summarizes the results for this experiment.

Results first illustrate that using the base network directly without adaptation (0 shot) does not show high performances and is not better than random noise, which means transfer learning is decisive. The adapted network also poorly performs when provided with one shot, but already indicates excellent potential with 4 or 5 shots. It is almost as effective as if the complete network was fully trained from scratch and tested on the same map. However, we note that the network slightly overfits the small subset of examples when given too many shots (ten and above).

3.4.3 Transfer Learning Rate

The transfer learning rate controls how fast the weights of the network will converge when fed with new configurations. It is essential to tune this hyperparameter correctly, as there are only a few shots and epochs available. The key is to find a balance between underfitting, overfitting, converging, and diverging.

In this section, we set the number of shots to 5 and froze the first four layers. The results are shown in figure 3.7.

Low transfer learning rate values (0.001 and below) cause underfit issues, and high values lead to severe divergence (1 and above). The best value found is 0.01.

Global results reveal in the first place that our convolutional network built to recognize spatial patterns of behavior on projected bitmaps does perform effectively in the navigation domain. While this may not be the case in most environments where the temporal aspect is highly valuable, it is already interesting to notice how a different data representation, combined with an appropriate handling strategy, does not affect the prediction quality and can even compete with state-of-the-art literature results.

Moreover, our experimental benchmarks aimed to shrink the gap between synthetic and real data using only raw 512x512 pixels maps whose structure is akin to images. Hence, no bias is introduced by an expert exhibiting complex handcrafted rules and reconstructing a model of the environment, often impossible with real-world data.

However, applying our technique in real-life settings still requires a considerable amount of resources to collect sufficient data to train the base network in the first place.

3.5 Conclusion

We presented a few-shot transfer learning approach for goal recognition in the navigation domain, exploiting a new spatial trail representation with a convolutional network and providing only a few examples for fast weights adaptation. Our few-shot transfer learning method demonstrated great potential in a specific context of long-term behaviors, indeed assisting a standard deep-learning architecture in both generalizing and reasoning with visual features. We additionally experimented with high-resolution bitmaps as a step toward operating on real data.

We furthermore want to share the incentive that our method could be applicable in a variety of domains, real or synthetic, by identifying a shared, intermediate

knowledge structure in the available data.

3.6 Acknowledgements

We thank *Compute Canada* for the computing resources they provided to support our research project.

References

[1] L. Amado, J. P. Aires, R. F. Pereira, M. C. Magnaguagno, R. Granada, F. Meneguzzi, "LSTM-Based Goal Recognition in Latent Space," *CoRR*, vol. abs/1808.05249, 2018. [Online]. Available: http://arxiv.org/abs/1808.05249

[2] M. Asai A. Fukunaga, "Classical Planning in Deep Latent Space: Bridging the Subsymbolic-Symbolic Boundary," in *AAAI 2018*, 2018. [Online]. Available: https://aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16302

[3] D. Avrahami-Zilberbrand G. A. Kaminka, "Fast and Complete Symbolic Plan Recognition," in *IJCAI 2005*, 2005, pp. 653–658.

[4] F. Bisson, H. Larochelle, F. Kabanza, "Using a Recursive Neural Network to Learn an Agent's Decision Model for Plan Recognition," in *IJCAI 2015*, 2015, pp. 918–924.

[5] H. H. Bui, S. Venkatesh, G. A. W. West, "Policy Recognition in the Abstract Hidden Markov Model," *JAIR*, vol. 17, pp. 451–499, 2002.

[6] E. Charniak R. P. Goldman, "A Bayesian Model of Plan Recognition," *Artif. Intell.*, vol. 64, no. 1, pp. 53–79, 1993. [Online]. Available: https://doi.org/10.1016/0004-3702(93)90060-O

[7] C. Chen, X. Zhang, S. Ju, C. Fu, C. Tang, J. Zhou, X. Li, "AntProphet: an Intention Mining System behind Alipay's Intelligent Customer Service Bot," in *IJCAI 2019*, 08 2019, pp. 6497–6499.

REFERENCES

[8] R. G. Freedman S. Zilberstein, "Integration of Planning with Recognition for Responsive Interaction Using Classical Planners," in *AAAI*, 2017.

[9] C. W. Geib R. P. Goldman, "A probabilistic plan recognition algorithm based on plan tree grammars," *Artificial Intelligence*, vol. 173, no. 11, pp. 1101–1132, 2009.

[10] R. Granada, R. Pereira, J. Monteiro, R. Barros, D. Ruiz, F. Meneguzzi, "Hybrid Activity and Plan Recognition for Video Streams," in *AAAI 2017*, 2017.

[11] D. Hana, Q. Liu, W. Fan, "A New Image Classification Method Using CNN transfer learning and Web Data Augmentation," *Expert Systems with Applications*, vol. 95, 11 2017.

[12] K. He, X. Zhang, S. Ren, J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," *CoRR*, vol. abs/1502.01852, 2015. [Online]. Available: http://arxiv.org/abs/1502.01852

[13] Y. Jia, Y. Zhang, R. Weiss, Q. Wang, J. Shen, F. Ren, z. Chen, P. Nguyen, R. Pang, I. Lopez Moreno, Y. Wu, "Transfer Learning from Speaker Verification to Multispeaker Text-To-Speech Synthesis," in *Advances in Neural Information Processing Systems 31* S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett, editors. Curran Associates, Inc., 2018, pp. 4480–4490. [Online]. Available: http://papers.nips.cc/paper/7700-transfer-learning-from-speaker-verification-to-multispeaker-text-to-speech-synthesis.pdf

[14] F. Kabanza, J. Filion, A. R. Benaskeur, H. Irandoust, "Controlling the Hypothesis Space in Probabilistic Plan Recognition," in *IJCAI 2013*, 2013, pp. 2306–2312.

[15] S. Keren, A. Gal, E. Karpas, "Goal Recognition Design with Non-Observable Actions," in *AAAI 2016*, 2016. [Online]. Available: https://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12222/12074

[16] D. P. Kingma J. Ba, "Adam: A Method for Stochastic Optimization," *CoRR*, vol. abs/1412.6980, 2014.

[17] Y. Kong Y. Fu, "Human Action Recognition and Prediction: A Survey," *CoRR*, vol. abs/1806.11230, 2018. [Online]. Available: http://arxiv.org/abs/1806.11230

[18] O. Köpüklü, M. Babaee, S. Hörmann, G. Rigoll, "Convolutional Neural Networks with Layer Reuse," *CoRR*, vol. abs/1901.09615, 2019. [Online]. Available: http://arxiv.org/abs/1901.09615

[19] K. Liu, W. Liu, C. Gan, M. Tan, H. Ma, "T-C3D: Temporal Convolutional 3D Network for Real-Time Action Recognition," *AAAI*, 2018. [Online]. Available: https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17205/16305

[20] P. Masters S. Sardiña, "Cost-Based Goal Recognition for Path-Planning," in *AAMAS 2017*, 2017, pp. 750–758.

[21] P. Masters S. Sardiña, "Cost-Based Goal Recognition in Navigational Domains," *JAIR*, vol. 64, pp. 197–242, 2019. [Online]. Available: https://doi.org/10.1613/jair.1.11343

[22] W. Min, B. W. Mott, J. P. Rowe, B. Liu, J. C. Lester, "Player Goal Recognition in Open-World Digital Games with Long Short-Term Memory Networks," in *IJCAI 2016*, 2016, pp. 2590–2596.

[23] W. Min, B. Mott, J. Rowe, R. Taylor, E. Wiebe, K. Boyer, J. Lester, "Multimodal Goal Recognition in Open-World Digital Games," in *AAAI 2017*, 2017. [Online]. Available: https://aaai.org/ocs/index.php/AIIDE/AIIDE17/paper/view/15910

[24] S. J. Pan, J. T. Kwok, Q. Yang, "Transfer Learning via Dimensionality Reduction," in *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2*, ser. AAAI'08. AAAI Press, 2008, pp. 677–682. [Online]. Available: http://dl.acm.org/citation.cfm?id=1620163.1620177

[25] R. F. Pereira, N. Oren, F. Meneguzzi, "Landmark-Based Heuristics for Goal Recognition," in *AAAI 2017*, 2017, pp. 3622–3628.

[26] R. F. Pereira, M. Vered, F. Meneguzzi, M. Ramírez, "Online Probabilistic Goal Recognition over Nominal Models," in *IJCAI 2019*, 2019, pp. 5547–5553. [Online]. Available: https://doi.org/10.24963/ijcai.2019/770

[27] M. Ramírez H. Geffner, "Probabilistic Plan Recognition Using Off-the-Shelf Classical Planners," in *AAAI 2010*, 2010.

[28] S. Ravi H. Larochelle, "Optimization as a Model for Few-Shot Learning," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017. [Online]. Available: https://openreview.net/forum?id=rJY0-Kcll

[29] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, D. Hassabis, "Mastering the Game of Go with Deep Neural Networks and Tree Search," *Nature*, vol. 529, no. 7587, pp. 484–489, January 2016.

[30] S. Sohrabi, A. V. Riabov, O. Udrea, "Plan Recognition As Planning Revisited," in *IJCAI 2016*, ser. IJCAI'16. AAAI Press, 2016, pp. 3258–3264. [Online]. Available: http://dl.acm.org/citation.cfm?id=3061053.3061077

[31] Y. C. Song, H. A. Kautz, J. F. Allen, M. D. Swift, Y. Li, J. Luo, C. Zhang, "A Markov logic framework for recognizing complex events from multimodal data," in *ICMI 2013*, 2013, pp. 141–148.

[32] N. Sturtevant, "Benchmarks for Grid-Based Pathfinding," *Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 2, pp. 144 – 148, 2012. [Online]. Available: http://web.cs.du.edu/~sturtevant/papers/benchmarks.pdf

[33] G. Sukthankar, C. Geib, H. H. Bui, D. Pynadath, R. P. Goldman, *Plan, Activity, and Intent Recognition: Theory and Practice*, 1st edition. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2014.

[34] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, C. Liu, "A Survey on Deep Transfer Learning," in *Artificial Neural Networks and Machine Learning – ICANN 2018* V. Kůrková, Y. Manolopoulos, B. Hammer, L. Iliadis, I. Maglogiannis, editors. Cham: Springer International Publishing, 2018, pp. 270–279.

[35] M. Vered G. A. Kaminka, "Heuristic Online Goal Recognition in Continuous Domains," in *IJCAI 2017*, 2017, pp. 4447–4454.

[36] M. Vered, R. F. Pereira, M. C. Magnaguagno, G. A. Kaminka, F. Meneguzzi, "Towards Online Goal Recognition Combining Goal Mirroring and Landmarks," in *AAMAS 2018*, 2018, pp. 2112–2114. [Online]. Available: http://dl.acm.org/citation.cfm?id=3238089

[37] O. Vinyals, I. Babuschkin, J. Chung, M. Mathieu, M. Jaderberg, "AlphaStar: Mastering the Real-Time Strategy Game StarCraft II," 2019. [Online]. Available: https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/

[38] J. Wang, Y. Chen, S. Hao, X. Peng, H. Lisha, "Deep Learning for Sensor-based Activity Recognition: A Survey," *Pattern Recognition Letters*, 07 2017.

[39] T. Wen, Y. Miao, P. Blunsom, S. J. Young, "Latent Intention Dialogue Models," in *ICML 2017*, 2017, pp. 3732–3741. [Online]. Available: http://proceedings.mlr.press/v70/wen17a.html

[40] S. Yan, Y. Xiong, D. Lin, "Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition," *AAAI*, 2018. [Online]. Available: https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17135/16343

[41] W. Zhao S. Du, "Spectral–Spatial Feature Extraction for Hyperspectral Image Classification: A Dimension Reduction and Deep Learning Approach," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 8, pp. 4544–4554, Aug 2016.

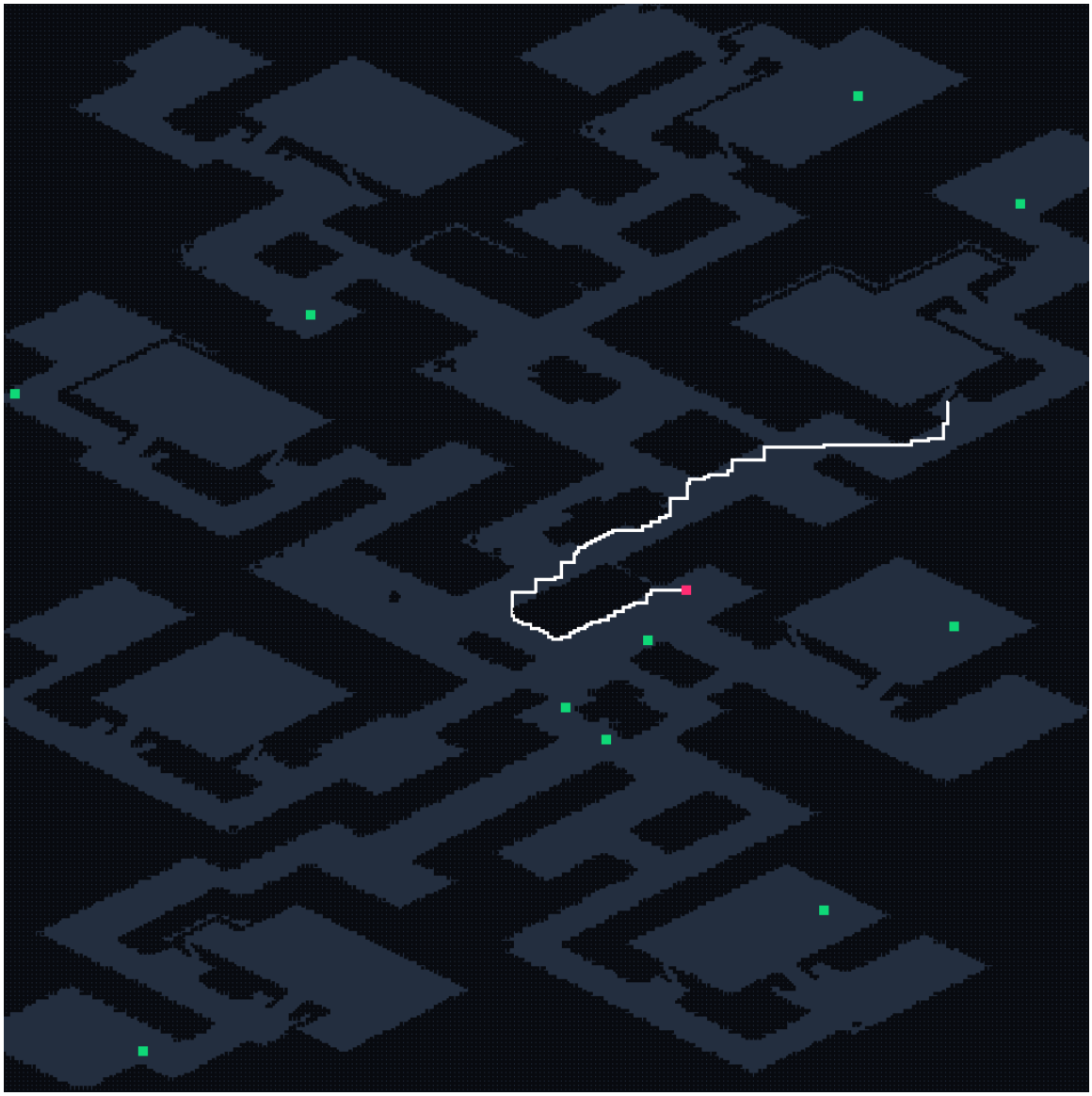

Figure 3.1 – An input example fed to the network (512x512 pixels). 5 channels represent either a wall (black), a free tile (gray), an observation (white), a start (red) or a goal (green). The path is here truncated at 75% of its total length. There are 10 possible goals.

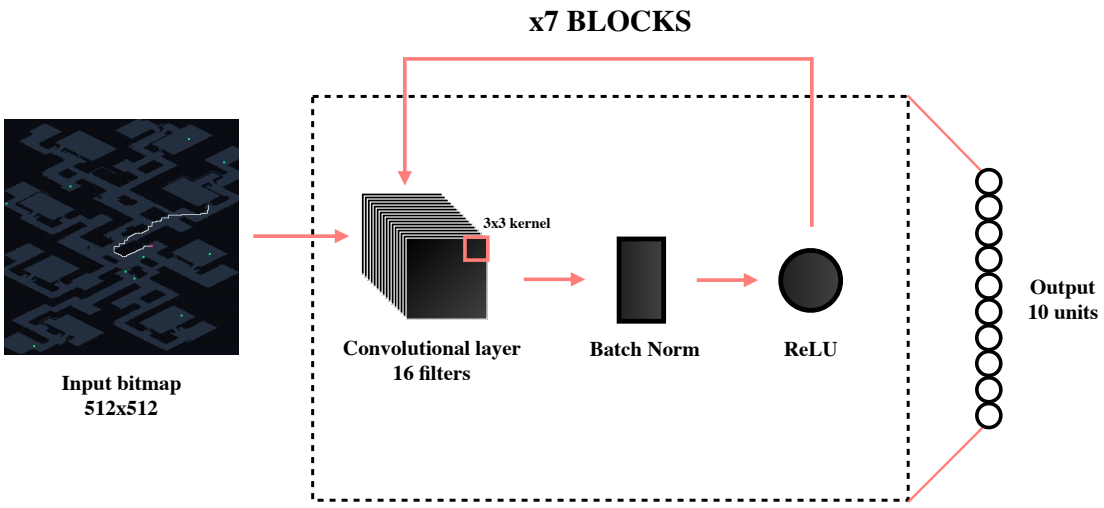

Figure 3.2 – The architecture of our network.

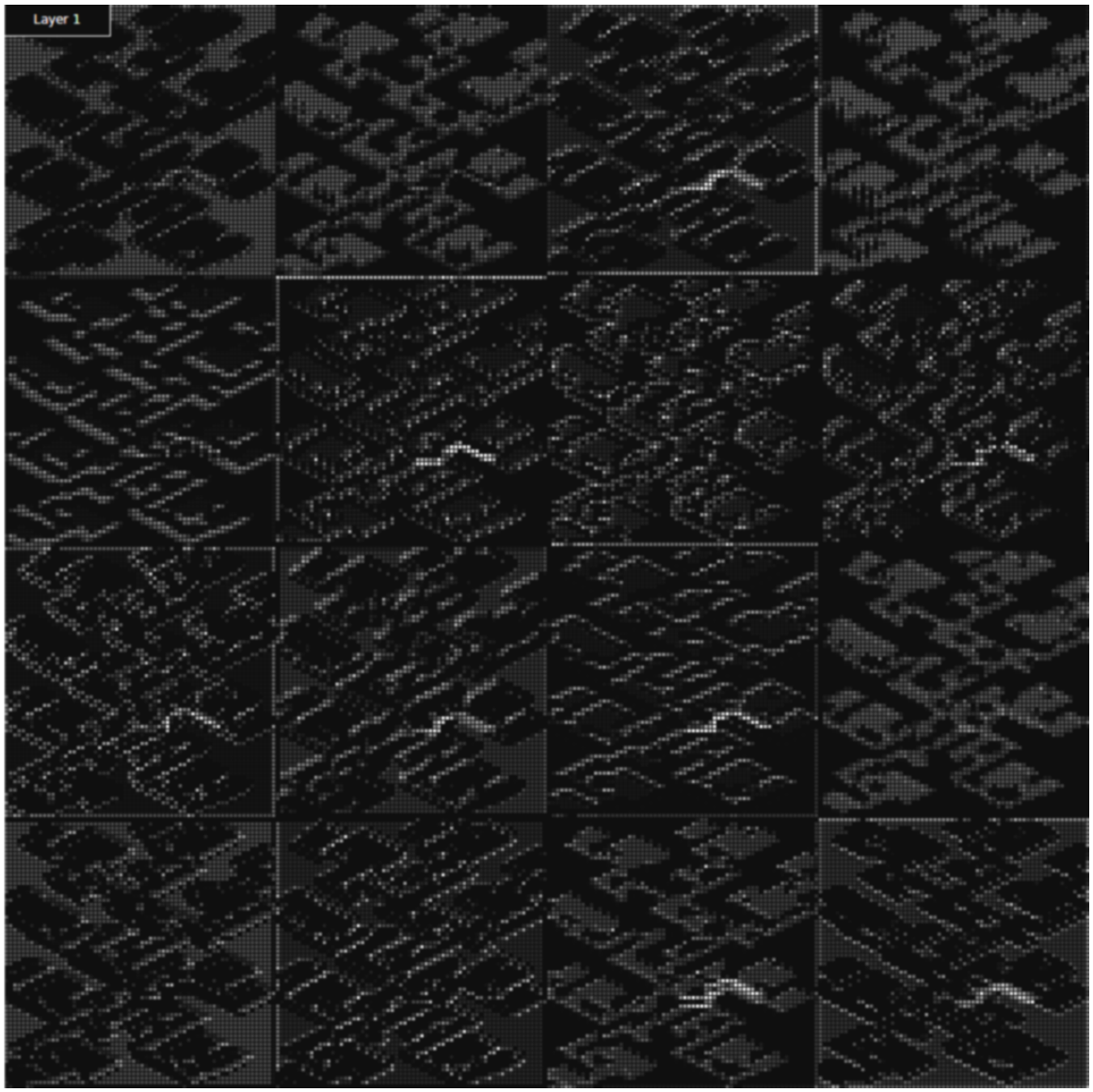

Figure 3.3 – The activation of the first convolutional layer (16 filters). There are clear patterns of edges, free tiles, obstacles and observations.

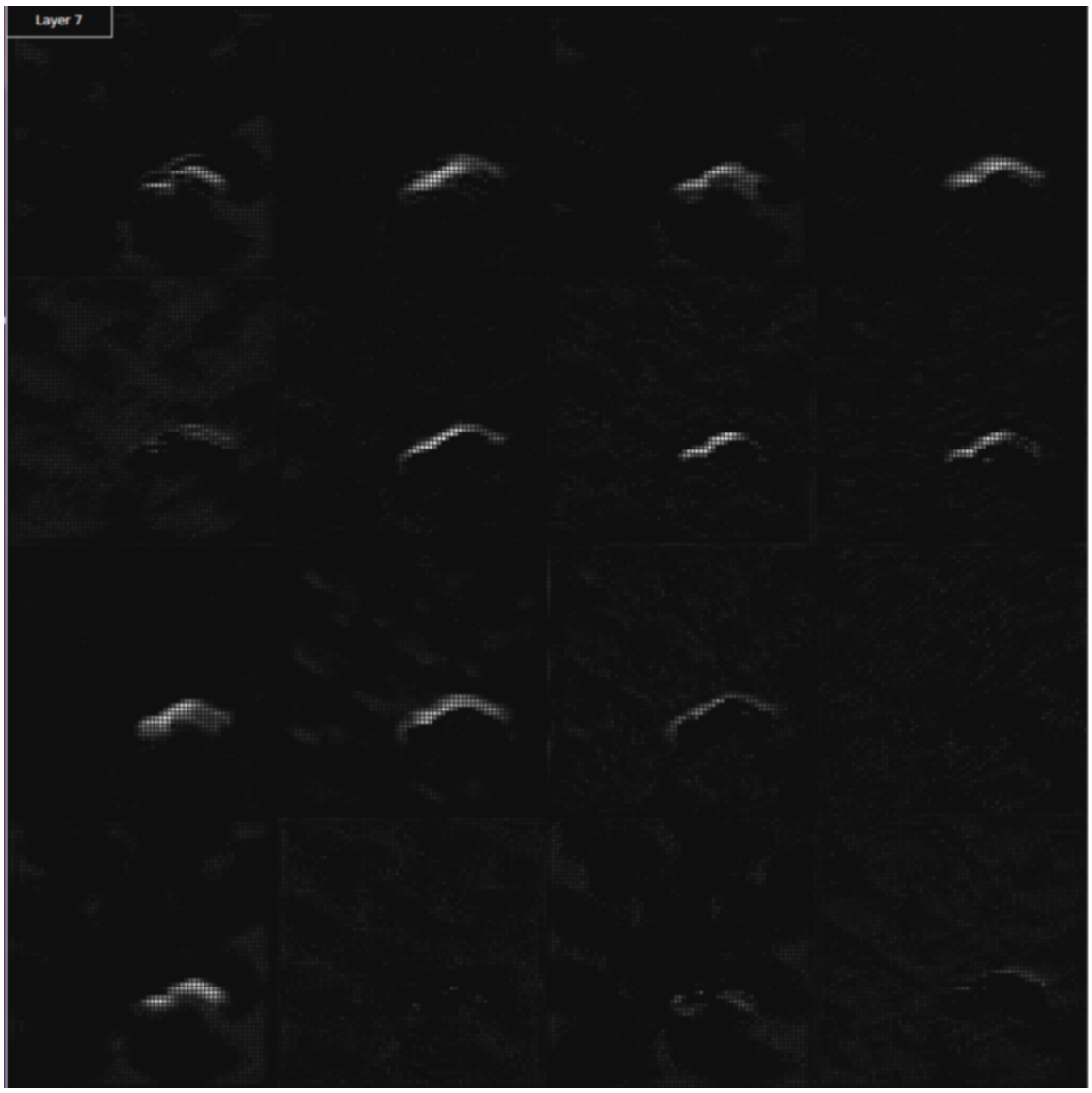

Figure 3.4 – The activation of the last convolutional layer (16 filters). The network is now focusing on the trajectory trail and highlighting observations that seems to be important to recognize the goal.

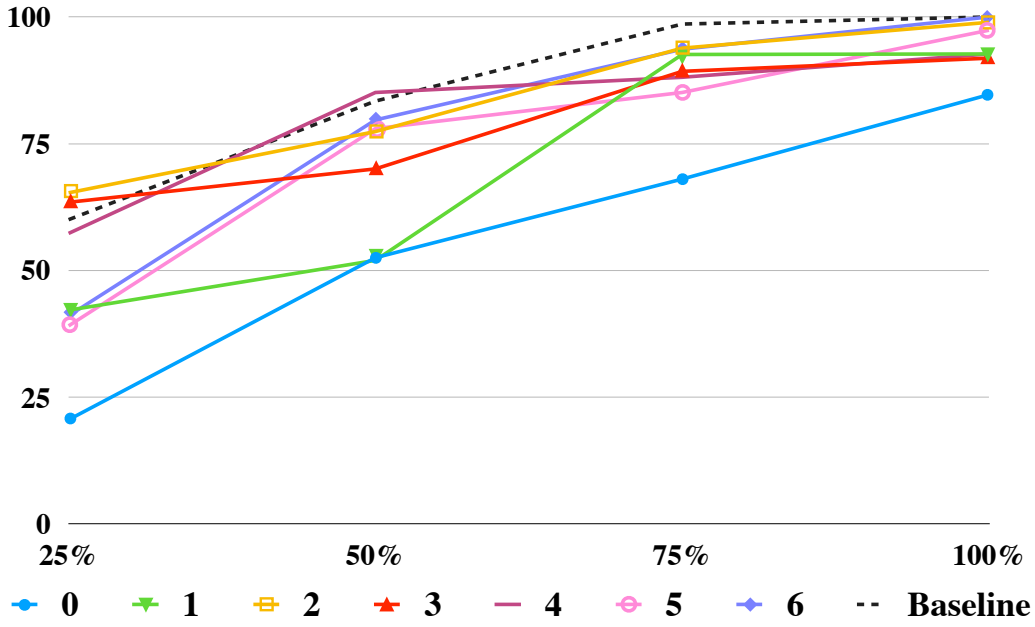

Figure 3.5 – Average test accuracy of a network adapted to five unseen configurations, depending on the number of locked convolutional layers. The x-axis designates the percentage of observations retained from the complete path. The baseline network, trained and tested on a single configuration, is shown with the dashed line.

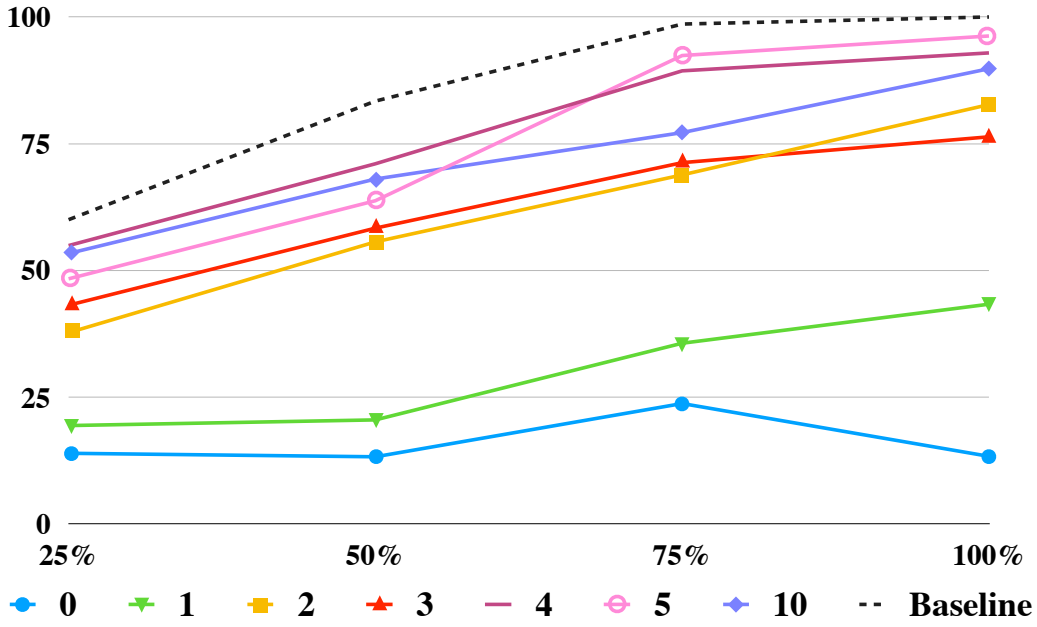

Figure 3.6 – Average test accuracy of a network adapted to five unseen configurations, depending on the number of shots provided. The x-axis designates the percentage of observations retained from the complete path. The baseline network, trained and tested on a single configuration, is shown with the dashed line.

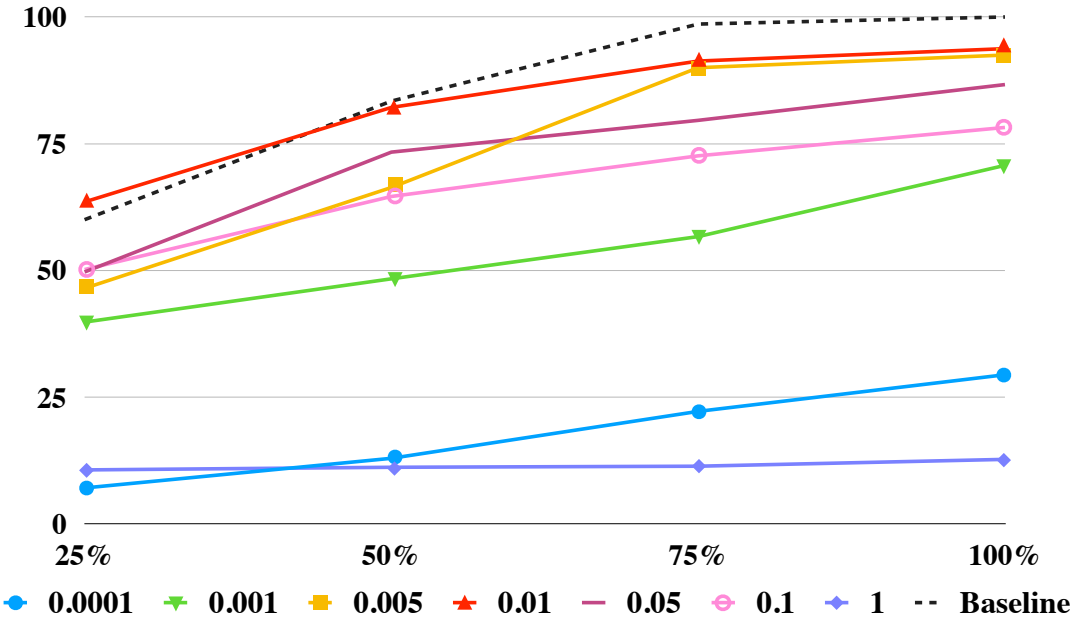

Figure 3.7 – Average test accuracy of a network adapted to five unseen configurations, depending on the transfer learning rate. The x-axis designates the percentage of observations retained from the complete path. The baseline network, trained and tested on a single configuration, is shown with the dashed line.

Chapitre 4

L'apprentissage profond avec une capacité d'imagination pour la reconnaissance de but

Les citations de ce chapitre redirigent vers les références à la page 92.

Résumé

Malgré les résultats encourageants de l'article précédent, l'approche reste limitée à des domaines spatiaux simples (comme le problème de navigation). De nombreuses approches [3, 7, 19] ont exploré les possibilités d'une fusion entre l'apprentissage profond et les connaissances symboliques pour bénéficier des avantages des deux paradigmes.

Nous proposons alors deux nouvelles méthodes permettant de généraliser l'apprentissage de la reconnaissance d'intention à des domaines similaires, en s'appuyant sur des métriques intermédiaires capables d'intégrer les décisions de l'agent d'une manière ou d'une autre. La première, appelée gradients de coûts, étend la portée des approches symboliques basées sur les coûts [22, 15] puisqu'elle permet d'encoder beaucoup plus d'informations à propos du comporte-

ment de l'agent et ne suppose pas un comportement rationnel de sa part, grâce à l'apprentissage profond. Elle est cependant coûteuse à mettre en place, ce qui nous a incité à créer la deuxième approche, nommée déviation séquentielle, qui approxime la première en utilisant une fonction heuristique estimant les coûts.

Les résultats de nos expérimentations démontrent que nos approches surpassent l'état de l'art dans la plupart des cas, que ce soit dans des domaines synthétiques de navigation ou avec des données réelles (analyse de l'intention de piétons à proximité d'un magasin [13]).

Commentaires

Cet article a été soumis à ICAPS (*International Conference on Automated Planning and Scheduling*) en 2020. Le projet a été mené conjointement par Thibault Duhamel et Mariane Maynard. Thibault a effectué les expérimentations pour la méthode des déviations séquentielles et a dirigé l'écriture de l'article. Mariane a effectué les expérimentations pour la méthode des gradients de coûts. Froduald Kabanza a supervisé les travaux et l'écriture de l'article.

Imagination-Augmented Deep Learning for Goal Recognition

Thibault Duhamel, Mariane Maynard, Froduald Kabanza
Département d'informatique
Université de Sherbrooke
Sherbrooke, Québec (Canada) J1K 2R1
thibault.duhamel@usherbrooke.ca,
mariane.maynard@usherbrooke.ca,
froduald.kabanza@usherbrooke.ca

Abstract

Being able to infer the goal of people we observe, interact with, or read stories about is one of the hallmarks of human intelligence. A prominent idea in current goal-recognition research is to infer the likelihood of an agent's goal from the estimations of the costs of plans to the different goals the agent might have. Different approaches implement this idea by relying only on handcrafted symbolic representations. Their application to real-world settings is, however, quite limited, mainly because handcrafted representations fail to capture well enough the factors that influence goal-oriented behaviors. In this paper, we introduce a novel idea of using a symbolic planner to compute plan-cost insights, which augment a deep neural network with an imagination capability, leading to improved goal recognition accuracy in real and synthetic domains compared to a symbolic recognizer or a deep-learning goal recognizer alone.

4.1 Introduction

Goal recognition is a fundamental cognitive ability, naturally performed by humans during their interactions. Often operating unconsciously, it is a crucial mechanism granting the possibility to foresee and integrate what may happen in the future to make better decisions according to additional projected information, either in cooperative or competitive environments. Artificially intelligent agents, however, still lack such powerful features despite recent breakthroughs in the field.

4.1. Introduction

One of the trending paradigms for implementing goal recognition algorithms relies on plan costs computed by a symbolic planner that inverses the planning process of the observed agent, leveraging the fact that they tend to act rationally towards their pursued goal [22, 15]. By computing plan cost differences, these methods indicate whether the agent is deviating from an optimal course to the goals and rank them according to this estimation. While promising, these approaches did not prove to be successful in real-world settings yet and still convey significant challenges. First, non-trivial tweaks are necessary to make reasonable inferences for situations where the optimality of the agents cannot be guaranteed [16]. Second, handcrafted representations used by the symbolic planner to compute expected plan costs may not be complete or precise, and symbolic planners are sensitive to such inaccuracies.

The gist of these approaches is that plan costs are good predictors of the goals pursued by the observed agents. They convey insight about which goals might be more demanding to achieve than others in the future, and require a planning process to derive them. This suggests we could use a deep learning method to learn to predict goals using plan costs as features. From this perspective, a deep neural network equipped with a planner to generate plan-cost features appears to be an imagination-augmented deep neural network [20]. Indeed, an imagination-augmented deep neural network can learn a policy from features generated by an *imagination* module, providing insight about the different futures that may occur if the agent takes any action in a set of possible ones.

Based on this analogy, we developed a novel approach to learn a goal-prediction model from features based on plan costs, with the idea that plan costs will convey insight about the future, improving the accuracy of a deep neural network compared to a baseline not using plan costs. On the other hand, given that plan costs are used as features of a learning algorithm, our hypothesis is that, unlike symbolic cost-based plan recognizer, our learned model would be more robust to errors in the representation used to compute plan costs, without requiring any tweaks to deal with situations where agents are not behaving optimally. We expect the model to automatically learn from data the extent to which an observed agent acts optimally in certain circumstances. We demonstrate the power of this novel idea by implementing two different methods to compute symbolic plan-cost-based features, respectively,

gradients of costs and *sequential deviations*. We show that each of them enables a deep neural network architecture to learn to better predict the goal of an observed agent than without such plan-cost-based features.

The rest of the paper is organized as follows: first, we provide key background concepts about the problem we solve. Then, we present our method, followed by the setup and results of our experiments. Finally, we provide a brief review of the literature related to our research work.

4.2 Background

Let us give a general definition of a goal recognition problem:

Definition 4.2.1 *A goal recognition problem is a tuple $\langle G, O \rangle$ where G is the set of possible goal states and $O = o_0, \ldots, o_t$ is the sequence of observations of an agent's behavior. O is generated from the interaction of the agent with an environment $E = \langle S, A, c \rangle$, composed of a set of states S, a set of actions $A : S \times S$ and a cost function $c : A \to \mathbb{R}_0^+$.*

In this paper, we assume full observability of the agent, i.e. we suppose we can fully extract E from O as well as $s_0, \ldots, s_t \in S$, the sequence of completely observed states, s_0 being the initial state. We also use the term *plan* to refer to a sequence of actions $a_0, \ldots, a_t \in A$ pursued by an agent, and $c(s_0, g)$ to define the cost of an optimal plan achieving $g \in G \subseteq S$ starting from s_0, where $c(s_0, g) = c(a_0) + c(a_1) + \ldots + c(a_t)$.

In this section, we provide background concepts explaining how to resolve this problem following two paradigms: cost-based goal recognition and goal recognition as learning.

4.2.1 Cost-Based Goal Recognition

The intuition behind cost-based goal recognition is that, assuming that the observed agent is rational (also known as cost-sensitive), they will be more likely to pursue the least costly plan. To perform goal inference, an observer only needs to compare the cost of the observed plan with the cost of an optimal plan for any given

goal, computed using an optimal planner over a domain theory of the environment. If the two costs match, then this goal is considered plausible [21].

Extending the inference with a probabilistic dimension is a mechanism partially coping for potential divergences from the optimal behavior. For instance, Ramírez and Geffner [22] compute the goal inference using a Boltzmann distribution:

$$P(g|O_{0:t}) = \alpha \frac{1}{1 + \exp(\beta \Delta(s_0, g, O_{0:t}))} \tag{4.1}$$

where α is a normalisation factor, β is a temperature hyperparameter tuned according to the agent's assessed optimality, and Δ is the following cost difference formula:

$$\Delta(s_0, g, O_{0:t}) = c(s_0, g, O_{0:t}) - c(s_0, g, \bar{O}_{0:t}) \tag{4.2}$$

where $c(s_0, g, O_{0:t})$ is the cost of an optimal plan from s_0 to g complying with the observed actions in $O_{0:t}$, and $c(s_0, g, \bar{O}_{0:t})$ is the cost of an optimal plan reaching g where at least one of the observed actions has not occurred.

Vered *et al.* [27] rather use a cost ratio to make a probabilistic inference:

$$P(g|O_{0:t}) = \alpha \frac{c(s_0, g)}{c(s_0, g, O_{0:t})} \tag{4.3}$$

Masters and Sardiña [15] use a simpler cost difference formula accounting only for the initial and last observations:

$$\Delta(s_0, s_t, g) = c(s_t, g) - c(s_0, g) \tag{4.4}$$

This method makes offline costs computing possible for some domains, such as the discrete navigation one, for which we can store the costs into convenient cost maps. They also suppose a Boltzmann probability distribution over this difference.

The ingenuity of cost-based goal recognition lies in the features used to compute goal inferences (optimal plan costs), which are quantities ranking the imagined future according to the agent's rationality.

However, computing a plan, even in the simple case of a deterministic environment under full observability, is NP-Complete [4]. These methods cannot be applied

realistically in situations where an agent needs to infer the goal of others quickly and where offline storage is not as trivial as in the navigation domain [15]. Approximated plan costs, computed by suboptimal planners or heuristic functions that run faster, can be used to infer an approximate distribution [21]. They are helpful in situations where the essential matter remains to identify the goals that are more likely.

Vered and Kaminka [26] introduced heuristics directly into the goal recognition inference process to judge whether a new observation changes the ranking of goals or whether a goal can be pruned, effectively reducing the number of calls to the planner.

Another work worth mentioning is the one of Sohrabi *et al.* [23], which computes the top-k optimal plans for each goal and adds a degree of compliance with the observations to their cost to deal with noisy and missing observations. Those additional quantities make it potentially more robust to suboptimal behaviors and errors in the model, but, in opposition to other works using suboptimal plan costs as predictors, it introduces a significantly higher computation overhead. Indeed, the method computes k times more plans than Ramírez and Geffner [22]'s technique, including both the optimal and suboptimal ones, and the value of k must be high to achieve comparable performance.

Other various studies present different ideas to reduce computation times using heuristic metrics instead of plan costs, with reduced accuracy. For instance, E.-Martín *et al.* [6] compute cost interaction estimates in plan graphs, while Pereira *et al.* [18] use landmarks, with the idea that goals with a higher completion ratio are more likely. We follow this line of inquiry by feeding one of our methods with heuristic metrics as an approximation of plan costs.

4.2.2 Goal Recognition as Learning

Another limit of previously presented methods lies in the inference algorithm, which relies exclusively on symbolic domain knowledge. If the knowledge happens to be incorrect, the algorithms may return inaccurate results. It thus becomes useful to have an adaptive inference process that can account for potential bias in the provided knowledge.

It is where learning algorithms intervene. The idea is to make an unbiased goal

inference directly from data by automatically extracting patterns from observed examples.

Given a set of goal recognition problems $\langle \mathcal{G}, \mathcal{O} \rangle$, let us assume that there exists an optimal probability function P that is maximal for a true goal $g^* \in G \in \mathcal{G}$, provided with the corresponding observations $O \in \mathcal{O}$, that is, $\text{argmax}_{g \in G} P(g|O) = g^*$.

Given the temporal nature of the sequence $O = o_0, \ldots, o_t$, this probability distribution can be approximated using a recurrent neural network such as a long short-term memory (LSTM) network:

$$P(g|O) \approx P'(g|O;\theta) = LSTM(O;\theta) = \text{softmax}(h_t)$$

where θ are the learned parameters of the network, and h_t is a transformation of O recursively defined as $h_t = \tanh(f(o_t, h_{t-1};\theta))$, where f is a transformation over o_t and h_{t-1} using θ.

Assuming we have access to a training dataset of paired examples (O, g^*) (i.e., we know the true goal g^* for a given $O \in \mathcal{O}$), we can train the set of parameters θ to minimize the number of erred predictions in our dataset of examples. In other words, we wish to minimize

$$L = \sum_{n=0}^{N} l(LSTM(O^n;\theta), g^* n)$$

where l is a loss function (such as the categorical cross-entropy) that is increasingly positive as $P'(g^*|O;\theta)$ approaches 0.

If the observations are non-symbolic, it can become useful to extract spatial information about the world as well using a spatiotemporal deep neural network (STDNN). In that case, we compute P' in the following manner:

$$P'(g|O;\theta) = STDNN(O;\theta) = LSTM(O';\theta)$$

where $O' = o'_0, \ldots, o'_t$ is a spatial-wise transformation of O using, for instance, convolutional layers in the case of grid-world navigation.

Some works explored LSTM networks trained on observed data with success for the task of goal recognition [17, 1]. However, these networks were trained and applied in single environment domains, and it is not realistic to expect they could generalize

to multiple environments.

It is where it becomes handy to explore plan-cost features, providing the model with cross-domain insight about the causal and long-term reasoning necessary to make informed goal inferences.

4.3 Method

We herein present our method as a combination of both paradigms, using neural networks fed by symbolic cost-based predictors. We introduce two novel features and approaches to learn from them.

4.3.1 Gradients of Costs (GC)

Previous works in cost-based goal recognition established that plan costs are undoubtedly good predictors. They tend, in fact, to suggest that at least two plan costs are necessary (one derived from the observed plan and another being non-contextual) and seem to be mandatory to make a comparison and assess the likelihood of the goal.

Considering the cost difference of Masters and Sardiña [15] given in equation 4.4, we observe that the cost $c(s_t, g^*)$ decreases as the agent completes its plan, increasing (in the negatives) the difference with $c(s_0, g^*)$. Therefore, it only makes sense that the probability of g^* increases as the difference widens.

In fact, if the behavior of the agent is purely rational, we can make the following observation:

Observation 1 *Let $s_0, \ldots, s_n$ be a sequence of observed states and g^* the true goal of the observed agent. Assuming their plan is optimal, then $c(s_t, g^*) \leq c(s_{t-1}, g^*) \forall t \in [1, n]$.*

Intuitively, the remaining cost of an optimal plan can only monotonically decrease as the agent advances towards their goal.

From this observation, we engineered a novel goal recognition feature by considering the partial derivative of an optimal cost over time. We compute the feature as

follows:

$$\frac{\partial c(s_t, g)}{\partial t} = c(s_{t-1}, g) - c(s_t, g) \tag{4.5}$$

where $c(s_t, g)$ is the optimal cost from the agent's state s_t to g [1]. By calculating the derivative for every possible $g \in G$, we obtain a vector of partial derivatives that we define to be the *gradient* of costs (GC) at t:

$$GC(s_t) = \left[\frac{\partial c(s_t, g)}{\partial t}\right]_{g \in G} \tag{4.6}$$

In other words, the vector $GC(s_t)$ gives a global idea about the current *moving direction* of the agent and which goal states are *towards* their move. Having a sequence of observed states $s_0, \ldots, s_n$, we can compute this quantity at multiple points in time. Using these as predictors and with the right inference algorithm, we can, in fact, obtain a goal recognition algorithm as effective as Masters and Sardiña [15]'s to evaluate rational behavior.

The interesting part appears to be its potential to make *better* goal inferences than Masters and Sardiña [15] and Ramírez and Geffner [22] for apparent *irrational* behavior. Indeed, making inferences over multiple cost differences instead of a single one saves more information about the observations, hence allowing more flexibility. This process is crucial to keep the system robust against certain misbeliefs conveyed by the domain knowledge.

Let us consider the example depicted in figure 4.1, where an agent navigates in a specific environment. The agent's behavior is suboptimal for all the goals since O is not on any optimal path to them [2]. Yet, this situation could realistically happen, if we imagine that the agent changed their mind, if some paths are less desirable than others, or if there are unseen obstacles. The point is, any misbelief conveyed by the knowledge we have about the world and the agent (deterministic, fully observable, uniform costs) can become problematic when the inference algorithm is fixed precisely over an engineered quantity.

1. Since we consider discrete timesteps, we approximate the partial derivative for a single timestep delta. We use the previous point $t - 1$ so that the formula does not depend on future information.

2. Following the definitions introduced by Masters and Sardina [16], the agent is strictly less rational, but not uniformly less rational.

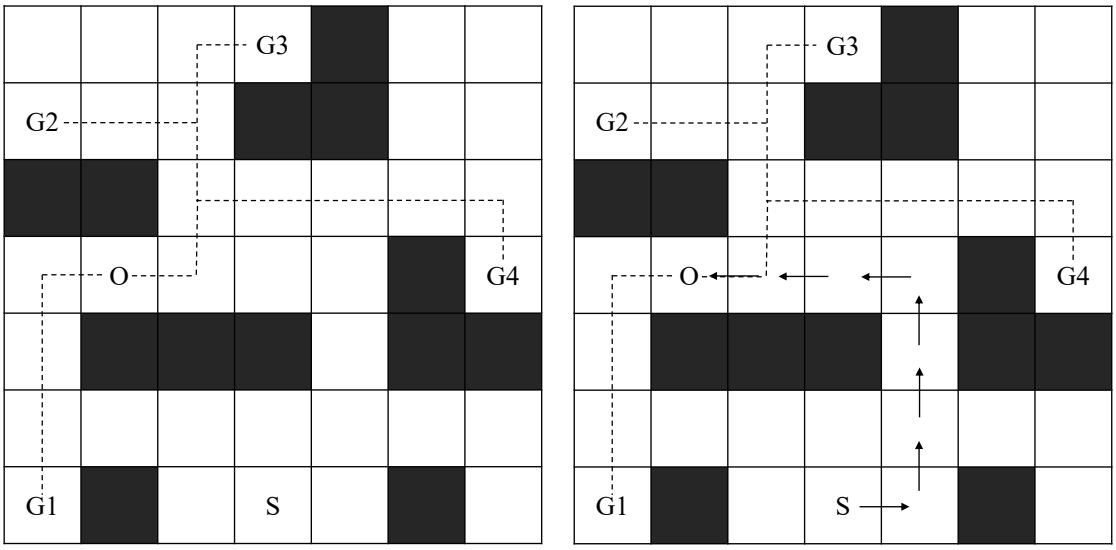

Figure 4.1 – Example of a suboptimal agent navigating in a grid. S is their initial position, O is their last observed position, G1 to G4 are potential goals, and the dashes are imaginary optimal paths. (Left) Without the observations in-between S and O, it is unclear what the destination of the agent is. (Right) With all the observations (arrows), G1 now appears as a likely goal.

Indeed, the information conveyed by the cost differences of equations 4.2, 4.3 and 4.4 is here ambiguous and lead to counter-intuitive results. For instance, equation 4.4 ranks both G2 and G3 first (since equation 4.1 is maximal when Δ is minimal) followed by both G1 and G4. Since it relies only on two observations to make an inference (as depicted on the left pane of 4.1), crucial information residing in the other observations do not weigh in the decision. Indeed, looking at the right pane and knowing that the agent started a loop, it now seems reasonable to consider G1 as more likely than G4.

Yet interestingly enough, equations 4.2 and 4.3 make the same prediction, even though their cost formulas rely on all observations. The reason is that they reduce the information conveyed in the observations to only two optimal costs for each goal. On the other hand, it is possible to use GC at multiple points in time to weigh each gradient feature according to their position in the sequence. Equations 4.2, 4.3 and 4.4 do not allow to do it in the time dimension, since they always compare to the initial

projected future at s_0, while equation 4.6 at point t is only function of the two last timesteps. The first values of $GC(s_t)$ do not affect the latest ones.

Furthermore, using a learning approach such as an LSTM network or an STDNN, it is possible for the learner to *forget* past gradient values by giving them a smaller weight if it helps it to cope with the agent's apparent suboptimality by not taking into account early observations that seem incorrect. All the same, the past gradient values can serve to avoid discarding a goal too early. Though produced by potentially inaccurate domain knowledge, gradients of costs contain all necessary information for our goal inference solution to balance the past and the future.

4.3.2 Sequential Deviations (SD): an Approximation of GC

While costs and gradients of costs convey meaningful information, they rely on expensive planners and a complete model of the environment. We explored the possibility to provide clues to a neural network but, this time, in the form of heuristic functions to lower computation costs.

A heuristic function is a function h that estimates the cost (or distance) of the optimal plan from a start state to a goal state. By extension, it can also take two states as parameters and compute an estimate of the distance between them. In the navigation domain, for instance, the L_2 (euclidean) distance is commonly used as a heuristic, since it represents the cost of perfect paths, from a bird's-eye view, for an unconstrained agent. In the rest of this paper, h will denote any heuristic function.

We first approximate the gradients of costs using h, such that we obtain a derivative:

$$\frac{\partial h(s_t, g)}{\partial t} = h(s_{t-1}, g) - h(s_t, g) \tag{4.7}$$

In the general case, the heuristic function does not decrease monotonically along the steps of an optimal path. However, if the heuristic is admissible (never overestimating the real optimal cost), we can apply the squeeze theorem and conclude that the heuristic will overall converge towards zero.

We introduce the *sequential deviation* (SD) metric, which estimates a temporal

deviation of an observed path O to every goal, defined as follows:

$$SD(O) = \left[\left[\frac{\partial h(s_t, g)}{\partial t} \right]_{t \leq |O|} \right]_{g \in G} \tag{4.8}$$

Although approximating GC, SD still illustrates the global motion of the agent. Looking at the figure 4.1 again, the L_2 heuristic function starts by increasing for G1, but then decreases. An inference algorithm attributing smaller weights to the past values would thus conclude that G1 is likely.

This method shows significant advantages. First of all, it allows bypassing the need for an environment model and planner in specific domains. Second, it is an approximation of the GC method, hence reducing the computation cost without losing the generalization capability. Moreover, the sequential deviation metric still encapsulates more temporal information than just the differences of costs from symbolic cost-based approaches.

4.4 Experiments

Although both methods could extend to task-planning problems, the experiments were limited to navigation benchmarks for now to allow a fair comparison with Masters and Sardiña [15]'s state-of-the-art algorithm. We begin by experimenting in a real-world setting, then conduct additional tests on arbitrarily complex navigation settings [15], to compare how incorrect models affect the predictions.

4.4.1 Pedestrians on a Crowded Street

UCY Zara [13] is a publicly available dataset of pedestrians walking in a crowded street near a store, made of CCTV video streams and 489 trajectories (sequences of coordinates), already identified from those images. We used 391 examples (80%) for training and saved 98 (20%) for testing. To run both our approaches and the baseline, we first adapted it to the goal recognition task by extracting a map from the video and determining the five main goals reached by those individual agents (store, left street, top right street, right street, bottom right street) from their last seen positions.

Figure 4.2(a) displays the five goals (in our experiments, we considered the centroid of each area to be the goal position) and figure 4.2(b) shows the environment extracted from the video, along with an example of a path from a real person. It is clear that captured behaviors do not follow optimal navigation patterns. Moreover, the obstacles may be incorrect, which would challenge the robustness of every approach.

While plans were computed using the A^* algorithm for GC and MS [15], costs were estimated with $h = L_2$ for SD. We used the same learning architecture and hyperparameters for both methods: as for the structure, depicted in figure 4.3, the encoders are LSTM networks with 64 units each and the dense layer outputs one unit per goal. All the weights are initialized using a uniform *He* distribution [9] and the output is softmax-activated. Since we solve a classification problem, we use the cross-entropy loss function to optimize our network with the Adam algorithm [11], whose learning rate is set to 0.001, $\beta_1 = 0.9$ and $\beta_2 = 0.999$.

We also built a simple LSTM, using the same initializer and optimizer, to compare the performances of pure deep learning and deep learning augmented with imagination capabilities.

We provide experimental comparisons using the accuracy metric, which is the number of correct predictions over the total number of predictions. A prediction is said to be correct if its highest score corresponds to the ground truth goal. In case of ties, we randomly draw one of the highest scores.

We evaluate the methods at different observable points in time (25%, 50%, 75%, and 100%). We implement this by truncating our observed paths to the given percentage (for instance, with a path made of 100 observations points and an observability of 25%, only the first 25 steps are considered).

Results are shown in figure 4.4 and confirm our hypothesis. We trained a simple LSTM on sequences of coordinates (*LSTM obs* in the graph) to assess how the imagination capability contributes to our deep learning methods. First, the deep learning part of our architecture indeed takes into account apparently erratic behaviors, because we do not assume the level of rationality of the agents. Second, the model extracted from the videos may be incomplete or incorrect, which suggests our approaches are more resilient to erroneous environment knowledge. Finally, the imagination capability helps to improve the performance of GC and SD, compared to a

simple deep learning pipeline.

4.4.2 Arbitrarily Complex Navigation

The problem we solve herein is the one of an agent navigating in a grid-world, a benchmark currently used in the state-of-the-art literature [15]. It consists in 30 StarCraft maps from the MovingAI Lab website[3] [25] adapted for goal recognition purposes. The objective is here to infer the destination of an agent by observing a trajectory of their visited positions. There are four possible actions: move up, down, right, or left. We generated five random goals per map and downscaled them to different sizes to evaluate how the methods would perform on problems of increasing complexity. Though synthetic, we introduced suboptimality in the agent's behavior by generating its path with a modified version of A* to mimic a human-controlled route, with a certain chance to drop an optimal step and pick a non-optimal one, using what we define as an ϵ-over-estimating heuristic:

Definition 4.4.1 *An ϵ-over-estimating heuristic is a function that returns an admissible quantity h' with a chance of $1 - \epsilon$, and $h' + \delta$ otherwise, where $\epsilon \in [0, 1]$ and $\delta > 0$.*

It is crucial to note that our approaches are trained on a set of different map configurations (obstacles, start, goals) and tested on another set of different map configurations, never seen before, to show the generalization capability of our methods.

We trained the GC-augmented network on full grid observations. Each of the observations takes the form of an 8-channel bitmap bird view of the environment where each channel represents whether the grid cell is an obstacle, a walkable cell, the observed agent's position, or one of its possible goal destinations. Each goal is attributed to a different channel to make them distinctive from one another. We here provided GC features in the form of differential cost maps, with partial derivatives yielded for every position. The resulting matrix was concatenated with the last observation, making it a 9-channel image input.

3. https://movingai.com/benchmarks/sc1/index.html

Since the set of environments used was known and finite, it was possible to compute cost maps offline and to store them before training and testing. To do so, we passed the bitmaps to the breadth-first search (*BFS*) algorithm for every position to generate the remaining cost from them. The resulting cost maps were stored in 30 4-dimension tensors, where the axes represent the coordinates of the start and end positions. The process was repeated for all problem sizes.

The neural network architecture (figure 4.5) is composed of:

1. 3 convolutional layers (*CNN*) of 16, 32, and 64 3x3 filters respectively, with a stride of one, same padding, and each followed by a ReLU activation;
2. an optional 2x2 max-pooling layer for 64x64 problems in-between each convolutional layer;
3. a convolutional LSTM layer (*Conv LSTM*) consisting of 32 3x3 filters for the cell state;
4. a fully connected layer (*FC*) of 256 units over the flattened output of the LSTM cell;
5. a final densely connected layer of 5 units followed by a softmax activation for goal inference.

Dropout [24] with a drop rate of 0.1 was applied in-between each parametrized layer. The network was trained using the categorical cross-entropy loss for 400 epochs for 16x16 maps and 2000 epochs for 64x64 maps. Each epoch consists of 64 training iterations of mini-batches of size 32. For this benchmark, we generated the examples in parallel to the training process, so that the network may have never seen the same example twice (even in-between epochs). The validation and test sets consist of 160 and 3000 generated examples, respectively.

Finally, the network was optimized using the same initializer and optimizer as in the previous benchmark. Furthermore, the learning rate is gradually reduced by a factor of 0.9 every 10 epochs when a plateau in validation loss is detected, to a minimal value of 1e-5.

As for SD, we trained the same architecture described for the real-world benchmark, for 10 epochs and with 10 000 examples, also generated in parallel to the training process.

Basic Experiment

We first tested our methods on different classic problems with accurate models of the environments. Results for grids of size 16x16, 64x64 and 128x128 are shown respectively in figures 4.6, 4.7 and 4.8, for $\epsilon = 0.2$ and $\delta \in [0, 10]$. For small-sized problems, GC outperforms the state-of-the-art algorithm from Masters and Sardiña [15] (MS), but SD demonstrates lower accuracy values. Those may be due to complex configurations where paths to different goals are overlapping each other and where approximated metrics such as heuristics cannot fully explain the observed behaviors. However, GC is unable to scale efficiently to larger complexities, as seen in figure 4.7. We can explain this phenomenon by considering the rapidly increasing input size for longer paths. Indeed, a sequence of 64x64 observations is eight times bigger than the same sequence of 16x16 ones. The network architecture was not enough complex to learn with such a rich input, despite the max-pooling layers reducing its dimension. As a result, we could not train GC on 128x128 maps with our available computing resources. On the other hand, SD surpasses other techniques when given larger problems, since it may convey more information about the general temporal moving direction when given longer sequences.

We may explain this outcome by reasoning about the amount of information we provide to each method. Gradients of costs embed every single movement, which is why they produce precise predictions but are expensive to compute. Symbolic algorithms, to the contrary, are limited only to two costs per goal and therefore deprived of heavily cutoff information in the temporal dimension. Finally, sequential deviations seem to withhold sufficient clues about the temporal sequences, without needing to compute precise costs.

Robustness to Erroneous Models

We then experimented with erroneous representations of the environments. To implement this notion, we applied definition 4.4.1 to modify the costs of transitions when computing the required paths for both GC (in the future module) and Masters and Sardiña [15] (in formula 4.4), such that we add a random value $\delta' \in [0, 10]$ to the real transition cost with a chance of $\epsilon' \in [0, 1]$. SD, whose computation does not

require planners, is not affected by this process.

Results are shown in figure 4.9 for $\epsilon' = 0.2$ and $\epsilon' = 1$ and illustrate that our methods handle incorrect environments more efficiently. We believe that the learning part of our model is crucial to adapt to such misbeliefs. It is also interesting to note how heuristics are unconditional estimates that do not essentially depend on environment knowledge.

4.5 Related Work

Deep learning has proven more than efficient for unstructured data classification, such as images and raw sensor data. Consequently, numerous architectures were experimented for short-term activity analysis [14, 12]. However, they mainly focus on identifying immediate actions without considering a larger temporal scope.

Although deep learning has made tremendous inroads in various activity recognition domains, it is surprisingly underused for agents engaged in long-term planning processes. Only a few research works explored the horizon of recurrent networks for long-term goal recognition. Min *et al.* [17] made use of LSTM networks to recognize the goal of a player from sequences of interactions in the game of Crystal Island, displaying promising results. Amado *et al.* [1] assembled a working pipeline with existing tools (a dense auto-encoder network from Asai and Fukunaga [2] with an LSTM), casting the problem of goal recognition as a regression task in a latent space for small games like 8-puzzle or tower of Hanoi.

An attractive alternative for goal or plan recognition that we exploit ourselves lies in the combination of the learning paradigm with a symbolic one to automatically approximate some domain knowledge from observations that complement expert resources. Bisson *et al.* [3] created a deep architecture mimicking HTN plan libraries to tune probabilistic inference models for plan recognition automatically. Granada *et al.* [7] built a convolutional network to identify primitive actions from videos and combined it with the Symbolic Behavior Recognition (SBR) algorithm based on an HTN plan library to detect the goals achieved in a kitchen environment. Pereira *et al.* [19] introduced a manner to construct a nominal model of the environment (states and transition rules) by the use of a deep network and then perform goal recognition

using a cost-based inference algorithm. The difference between these works and ours is that our models learn directly from knowledge whose format is non-specific to any environment, making them transferable to multiple ones.

The future projection capability is seeing a growing research interest from the deep learning community. Imagination-augmented agents [20] that inspired our work go further by using model-based deep reinforcement learning ideas to imagine future projected trajectories to guide the exploration of a model-free deep policy learner in Sokoban, PacMan and other related games. Dosovitskiy and Koltun [5] transform the standard reinforcement learning setting into a self-supervised one by attempting to predict action effects on measurements (such as altitude, health). Ha and Schmidhuber [8] use variational auto-encoders to simulate world models from games and an evolutionary algorithm to learn from these simulations. Ke *et al.* [10] effectively learn to predict some long-term future using improved LSTM architectures and show how it helps in various planning tasks, either deep-learned by imitation or by reinforcement. While these approaches performed on multiple problems involving long-term reasoning, long-term goal recognition is not one of them. Another aspect is that they all chose to learn future projection, while we rely on symbolic models and planners. They enabled us to achieve impressive results on challenging problems using a simpler architecture and fewer data.

4.6 Conclusion

We presented two innovative solutions to goal recognition by combining imagination-augmented deep learning architectures with costs-based features derived from symbolic knowledge. The ability to project the observed agent into the future, which is inherent to long-term goal recognition, helps generalize to multiple configurations. The first metric, *gradients of costs*, encodes more temporal information that just a difference of costs, but is expensive. The second one, *sequential deviations*, helps reduce the computational cost by approximating the previous one with heuristic functions so that no planner is required anymore.

Our solution outperforms the state-of-the-art sheer symbolic methods, both in synthetic and real environments. We demonstrated that our approaches could more

efficiently predict the goal of the agent when our assumptions about their behavior are wrong (that is, when they are suboptimal and when the environment model is erroneous), hence proving its robustness.

4.7 Acknowledgements

We would like to thank *Compute Canada* for the computing resources they provided and the *NVIDIA Corporation* for the Quadro P6000 graphic card they donated.

References

[1] L. Amado, J. P. Aires, R. F. Pereira, M. C. Magnaguagno, R. Granada, F. Meneguzzi, "LSTM-Based Goal Recognition in Latent Space," *CoRR*, vol. abs/1808.05249, 2018. [Online]. Available: http://arxiv.org/abs/1808.05249

[2] M. Asai A. Fukunaga, "Classical Planning in Deep Latent Space: Bridging the Subsymbolic-Symbolic Boundary," in *AAAI 2018*, 2018. [Online]. Available: https://aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16302

[3] F. Bisson, H. Larochelle, F. Kabanza, "Using a Recursive Neural Network to Learn an Agent's Decision Model for Plan Recognition," in *IJCAI 2015*, 2015, pp. 918–924.

[4] G. F. Cooper, "The Computational Complexity of Probabilistic Inference Using Bayesian Belief Networks (Research Note)," *Artificial Intelligence*, vol. 42, no. 2-3, pp. 393–405, March 1990. [Online]. Available: http://dx.doi.org/10.1016/0004-3702(90)90060-D

[5] A. Dosovitskiy V. Koltun, "Learning to Act by Predicting the Future," in *ICLR 2017*, 2017. [Online]. Available: https://openreview.net/forum?id=rJLS7qKel

[6] Y. E.-Martín, M. D. R.-Moreno, D. E. Smith, "A Fast Goal Recognition Technique Based on Interaction Estimates," in *IJCAI 2015*, 2015, pp. 761–768.

[7] R. Granada, R. Pereira, J. Monteiro, R. Barros, D. Ruiz, F. Meneguzzi, "Hybrid Activity and Plan Recognition for Video Streams," in *AAAI 2017*, 2017.

[8] D. Ha J. Schmidhuber, "Recurrent World Models Facilitate Policy Evolution," in *NeurIPS 2018*, 2018, pp. 2450–2462. [Online]. Available: http://papers.nips.cc/paper/7512-recurrent-world-models-facilitate-policy-evolution.pdf

[9] K. He, X. Zhang, S. Ren, J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," *CoRR*, vol. abs/1502.01852, 2015. [Online]. Available: http://arxiv.org/abs/1502.01852

[10] N. R. Ke, A. Singh, A. Touati, A. Goyal, Y. Bengio, D. Parikh, D. Batra, "Modeling the Long Term Future in Model-Based Reinforcement Learning," in *ICLR 2019*, 2019. [Online]. Available: https://openreview.net/forum?id=SkgQBn0cF7

[11] D. P. Kingma J. Ba, "Adam: A Method for Stochastic Optimization," *CoRR*, vol. abs/1412.6980, 2014.

[12] Y. Kong Y. Fu, "Human Action Recognition and Prediction: A Survey," *CoRR*, vol. abs/1806.11230, 2018. [Online]. Available: http://arxiv.org/abs/1806.11230

[13] A. Lerner, Y. Chrysanthou, D. Lischinski, "Crowds by Example," *Comput. Graph. Forum*, vol. 26, pp. 655–664, 2007.

[14] K. Liu, W. Liu, C. Gan, M. Tan, H. Ma, "T-C3D: Temporal Convolutional 3D Network for Real-Time Action Recognition," *AAAI*, 2018. [Online]. Available: https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17205/16305

[15] P. Masters S. Sardiña, "Cost-Based Goal Recognition in Navigational Domains," *JAIR*, vol. 64, pp. 197–242, 2019. [Online]. Available: https://doi.org/10.1613/jair.1.11343

[16] P. Masters S. Sardina, "Goal Recognition for Rational and Irrational Agents," in *AAMAS 2019*, Richland, SC, 2019, pp. 440–448. [Online]. Available: http://dl.acm.org/citation.cfm?id=3306127.3331725

[17] W. Min, B. W. Mott, J. P. Rowe, B. Liu, J. C. Lester, "Player Goal Recognition in Open-World Digital Games with Long Short-Term Memory Networks," in *IJCAI 2016*, 2016, pp. 2590–2596.

[18] R. F. Pereira, N. Oren, F. Meneguzzi, "Landmark-Based Heuristics for Goal Recognition," in *AAAI 2017*, 2017, pp. 3622–3628.

[19] R. F. Pereira, M. Vered, F. Meneguzzi, M. Ramírez, "Online Probabilistic Goal Recognition over Nominal Models," in *IJCAI 2019*, 2019, pp. 5547–5553. [Online]. Available: https://doi.org/10.24963/ijcai.2019/770

[20] S. Racanière, T. Weber, D. P. Reichert, L. Buesing, A. Guez, D. J. Rezende, A. P. Badia, O. Vinyals, N. Heess, Y. Li, R. Pascanu, P. W. Battaglia, D. Hassabis, D. Silver, D. Wierstra, "Imagination-Augmented Agents for Deep Reinforcement Learning," in *NIPS 2017*, 2017, pp. 5690–5701. [Online]. Available: http://papers.nips.cc/paper/7152-imagination-augmented-agents-for-deep-reinforcement-learning

[21] M. Ramírez H. Geffner, "Plan Recognition as Planning," in *IJCAI 2009*, 2009, pp. 1778–1783.

[22] M. Ramírez H. Geffner, "Probabilistic Plan Recognition Using Off-the-Shelf Classical Planners," in *AAAI 2010*, 2010.

[23] S. Sohrabi, A. V. Riabov, O. Udrea, "Plan Recognition As Planning Revisited," in *IJCAI 2016*, ser. IJCAI'16. AAAI Press, 2016, pp. 3258–3264. [Online]. Available: http://dl.acm.org/citation.cfm?id=3061053.3061077

[24] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[25] N. Sturtevant, "Benchmarks for Grid-Based Pathfinding," *Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 2, pp. 144 – 148, 2012. [Online]. Available: http://web.cs.du.edu/~sturtevant/papers/benchmarks.pdf

REFERENCES

[26] M. Vered G. A. Kaminka, "Heuristic Online Goal Recognition in Continuous Domains," in *IJCAI 2017*, 2017, pp. 4447–4454.

[27] M. Vered, G. Kaminka, S. Biham, "Online goal recognition through mirroring: humans and agents," in *Fourth Annual Conference on Advances in Cognitive Systems*, 2016.

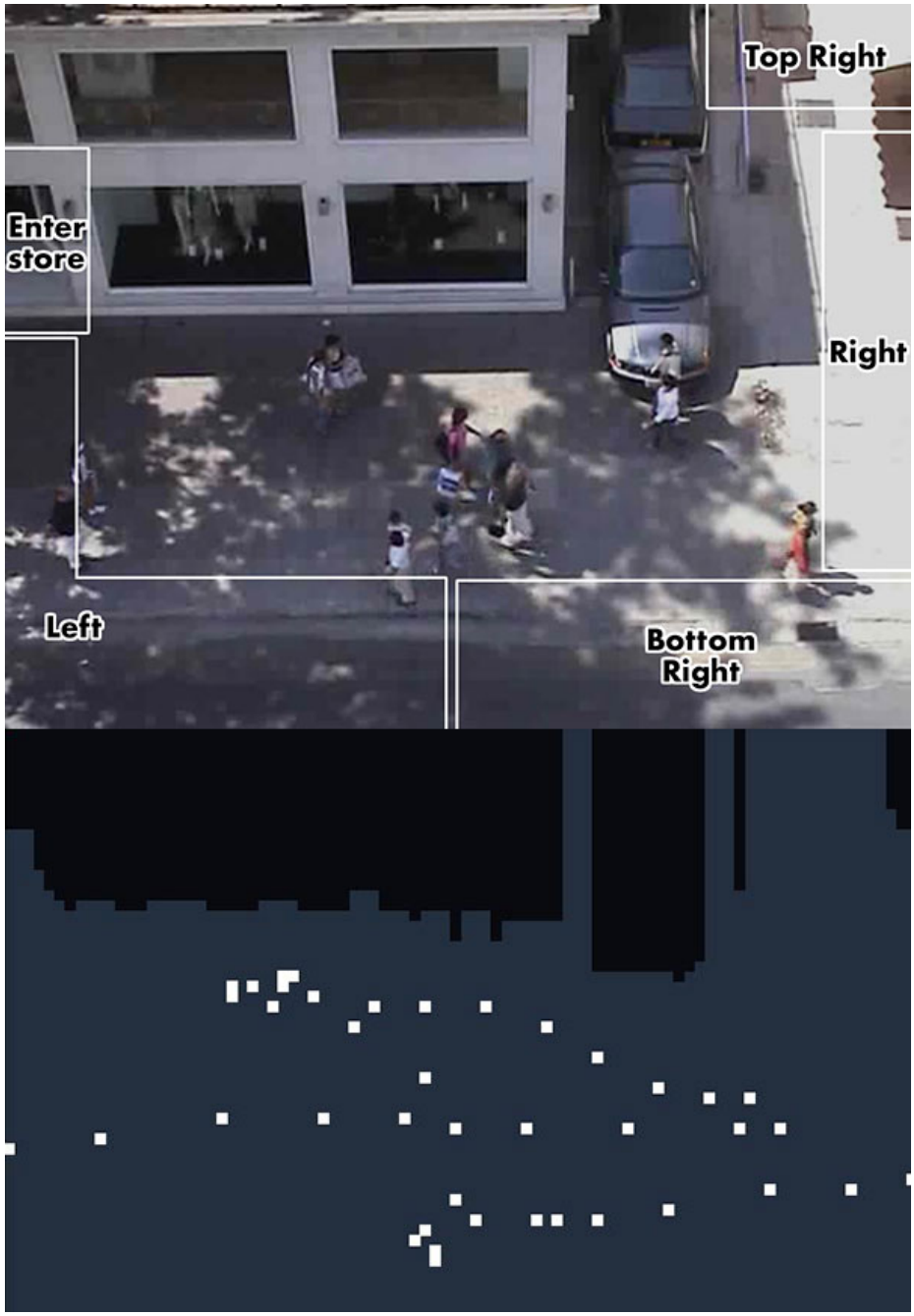

Figure 4.2 – (a) On the top, the different goals achieved by the pedestrians in the video. (b) On the bottom, the grid environment extracted from the raw video. The white dots represent the path of one person. It is clear that the observed behaviors are erratic.

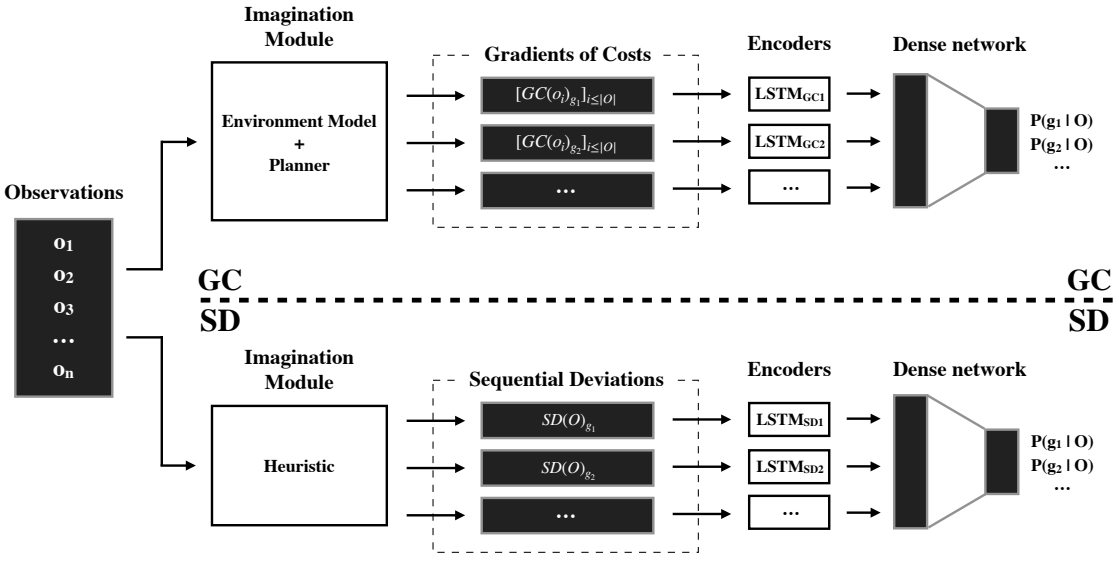

Figure 4.3 – GC (top-half) and SD (bottom-half) learning architectures for the UCY dataset.

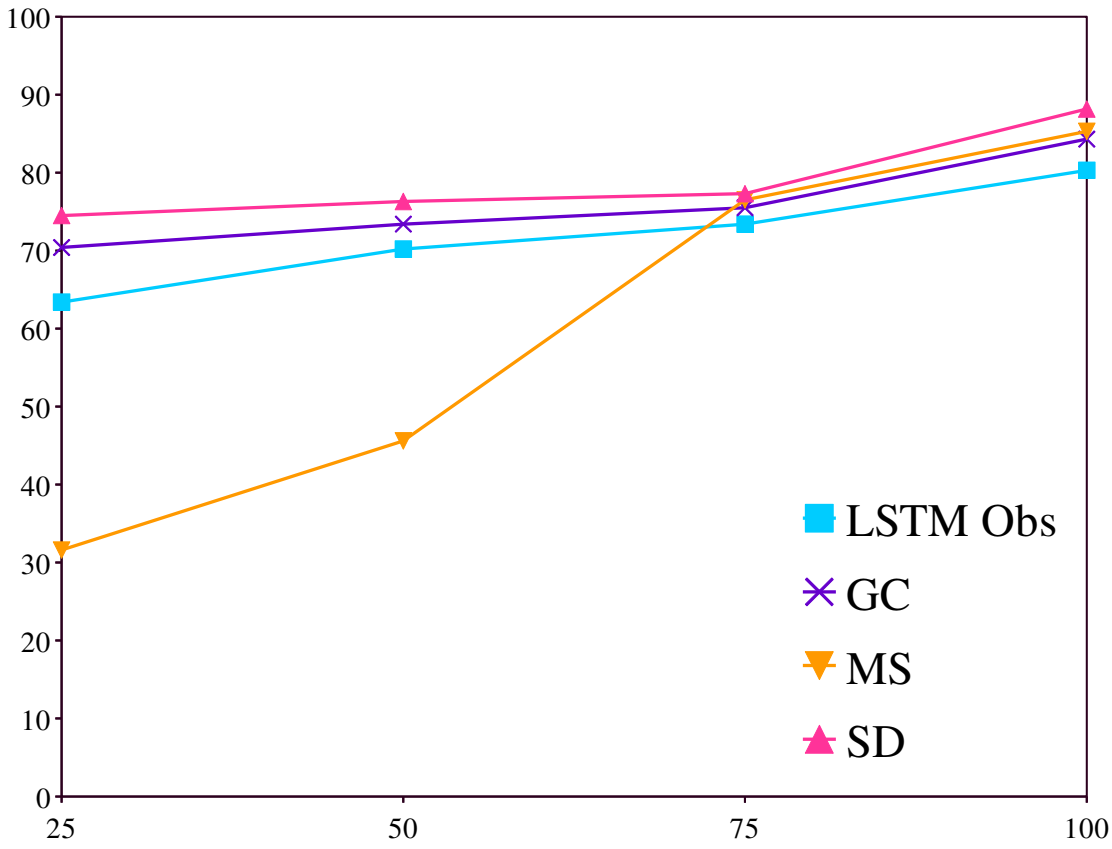

Figure 4.4 – Results on the UCY Zara dataset.

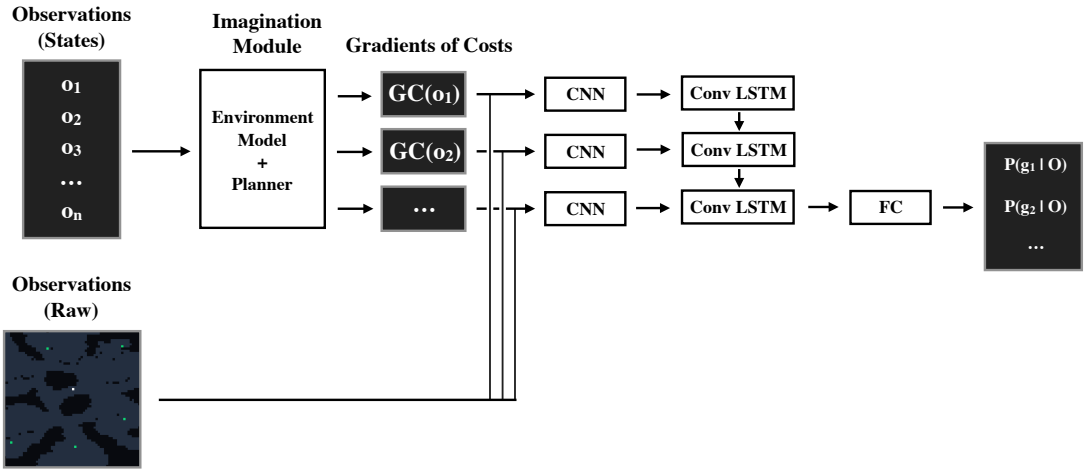

Figure 4.5 – GC network architecture for the navigation domain. The optional max-pooling layers are not displayed.

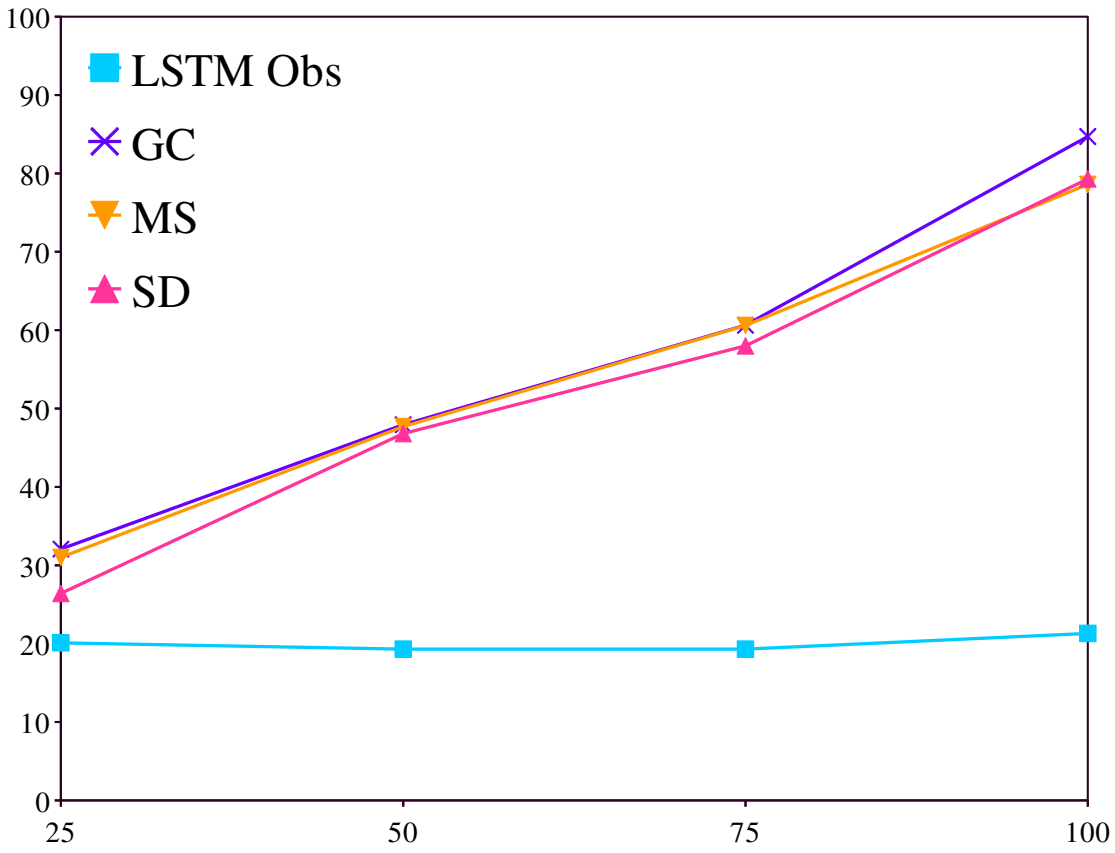

Figure 4.6 – Results on 16x16 grids.

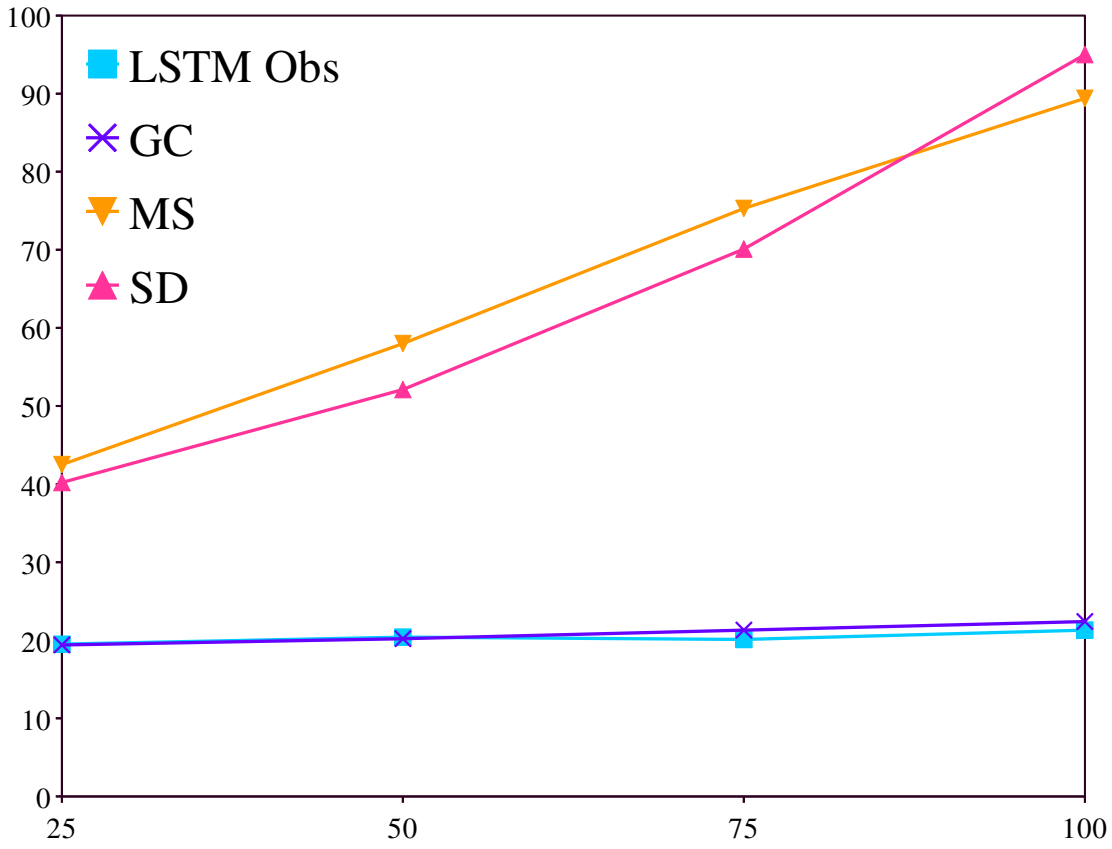

Figure 4.7 – Results on 64x64 grids.

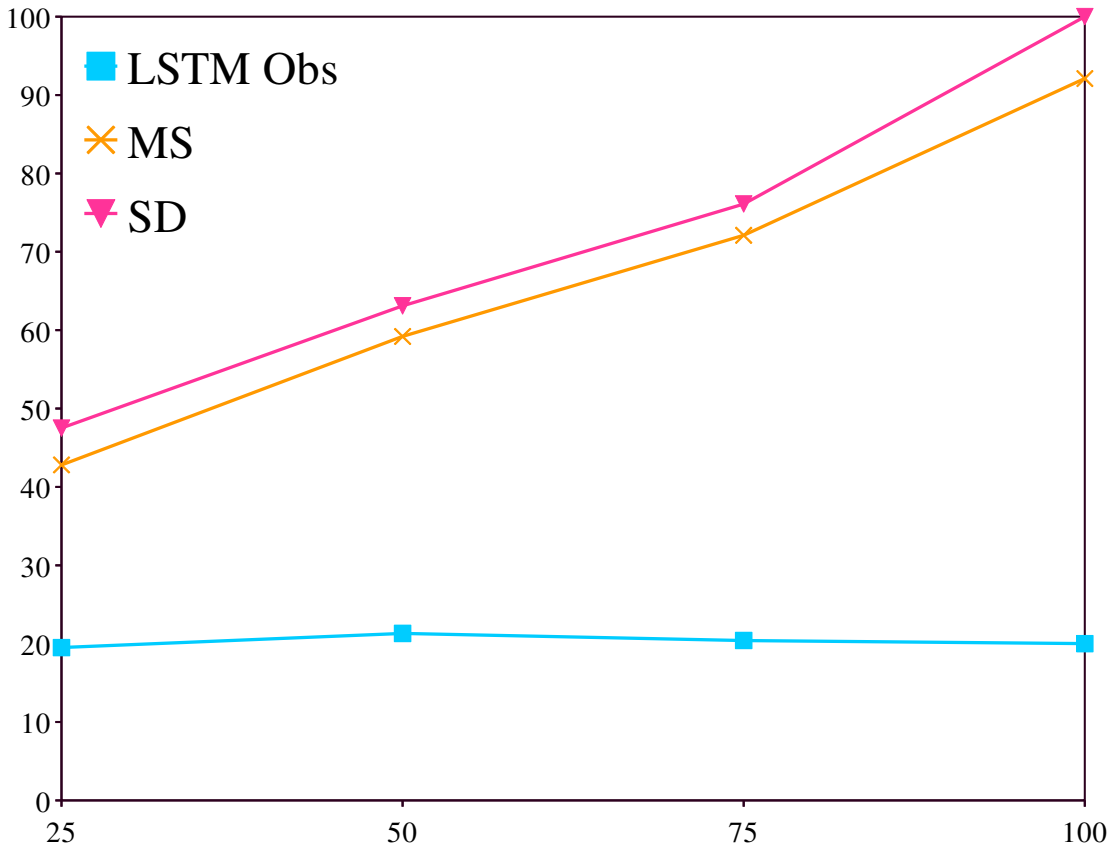

Figure 4.8 – Results on 128x128 grids.

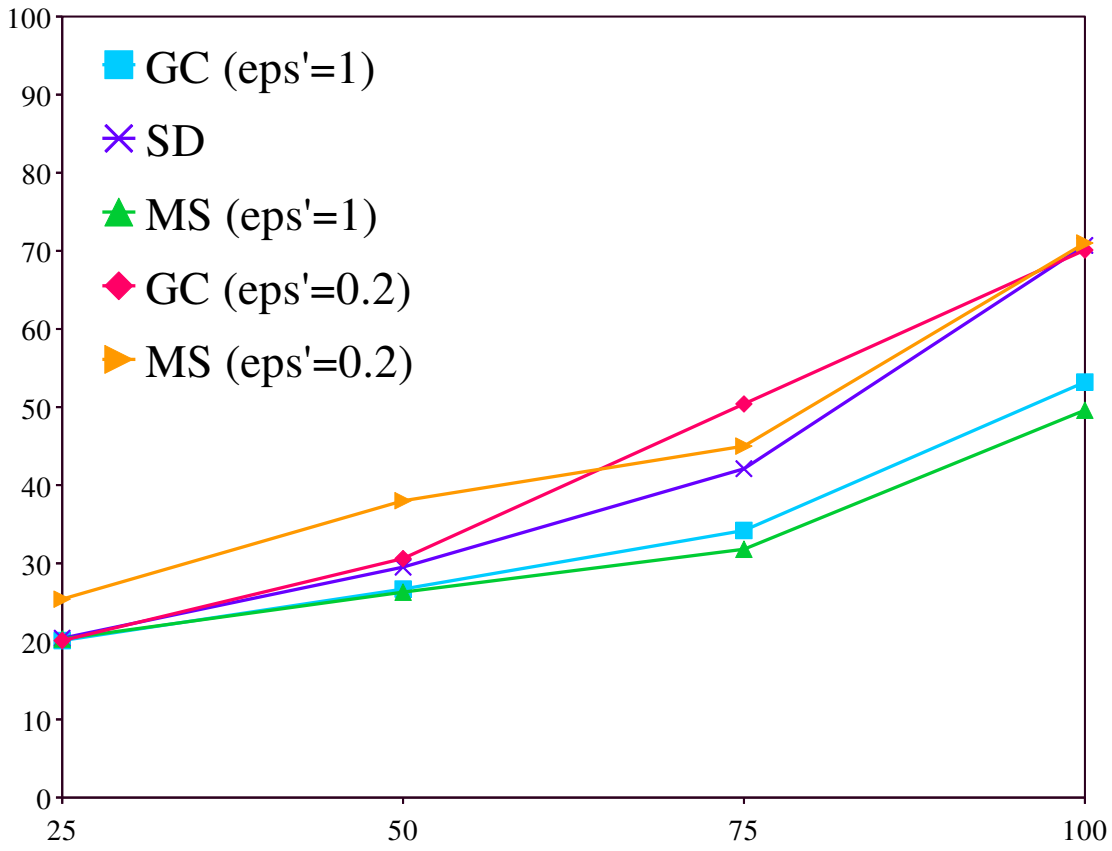

Figure 4.9 – Results for $\epsilon' = 0.2$ and $\epsilon' = 1$ (16x16 grid).

Conclusion

Les citations de la conclusion redirigent vers la bibliographie à la fin du mémoire (page 106).

Nous avons présenté dans ce mémoire trois articles scientifiques sur la reconnaissance de but, problème encore en cours de résolution, en se basant sur les capacités de l'apprentissage profond.

Le premier, proposant une comparaison de différentes architectures de réseaux de neurones avec les méthodes symboliques de l'état de l'art [46, 35], montre que ces dernières étaient dépassées en performance dans des contextes simples de navigation et de planification de tâches. Cela s'explique notamment par le fait que les approches symboliques s'appuient sur le principe de rationalité pour le comportement de l'agent, ce qui n'est pas respecté dans la majorité des situations, et qu'elles sont mises en défaut le cas échéant. En opposition à cela, l'apprentissage profond semble arriver à capter les variations du comportement sous-optimal de l'agent, mais ne possède pas la faculté de généraliser à plusieurs domaines.

Le deuxième article suggère alors une piste de réflexion pour permettre de généraliser l'apprentissage de la reconnaissance d'intention à plusieurs environnements similaires. S'inspirant des principes d'apprentissage avec peu d'exemples (*few-shot learning*) et de transfert d'apprentissage (*transfer learning*), il apporte l'alternative d'utiliser un réseau à convolution pour exploiter un espace intermédiaire spatial et de ré-entraîner uniquement les poids qui ne sont pas partagés entre les différentes configurations. Cependant, quoiqu'efficace, cette méthode reste limitée à des domaines relativement dépendants de la proximité spatiale, comme le problème de navigation.

Enfin, le troisième article introduit deux méthodes combinant l'apprentissage pro-

fond avec des quantités intermédiaires obtenues à partir de connaissances symboliques et les compare à l'état de l'art [35]. La première métrique, appelée gradients de coûts, repose sur la différence temporelle des coûts pour se rendre à chacun des buts. La deuxième, nommée déviation séquentielle, s'appuie sur une fonction heuristique pour estimer le coût entre deux états de l'agent. Les résultats démontrent que nos méthodes généralisent à plusieurs environnements et gèrent également les comportements erratiques d'individus dans un jeu de données réelles, comme souligné par la comparaison avec l'état de l'art symbolique basé sur les coûts.

L'étape suivante de ce travail de recherche consistera à apprendre à reconnaître l'intention dans des domaines réels à l'aide de données brutes. En effet, nos approches et celles de la littérature fonctionnent toutes actuellement sur des données pré-traitées (extraction manuelle ou automatique des coordonnées d'un piéton dans une vidéo, des actions effectuées par l'agent...), ce qui s'apparente, dans certains cas, à une perte d'informations. Dans le futur, nous souhaiterions ne pas nous limiter uniquement aux cadres définis par des modèles mais plutôt analyser automatiquement les informations pertinentes cachées dans des données à bas niveau, trop complexes à énumérer. Par exemple, au lieu de ne considérer que la position d'un agent dans une vidéo, il serait beaucoup plus efficace d'y reconnaître des composantes supplémentaires comme les objets/vêtements qu'il porte, les éléments du décor qui l'entourent, son attitude...

Finalement, cela revient à s'interroger sur l'accessibilité de l'information. Les approches symboliques, par exemple, supposent l'existence d'un mécanisme qui, à chaque action effectuée par l'agent, met à jour l'état de celui-ci [35] ou permet de retracer son historique [46]. Par conséquent, il faut étudier l'applicabilité de chaque méthode en adéquation avec le problème considéré en pratique, selon la nature du suivi des connaissances dont on peut disposer.

En outre, on note que le cadre de nos travaux se limite à l'observation d'un agent évoluant seul et sans pression dans son environnement. De nombreuses applications n'affichent pas les mêmes axiomes : c'est le cas par exemple des jeux-vidéos, où les joueurs cherchent à tromper leurs adversaires sur leurs propres stratégies [5], ou des cadres de sécurité, dans lesquels l'attaquant essaye de leurrer le système de surveillance. Il faudrait pour cela analyser dans quelle mesure l'apprentissage profond permettrait de détecter de tels comportements illusoires.

Bibliographie

[1] L. Amado, J. P. Aires, R. F. Pereira, M. C. Magnaguagno, R. Granada, F. Meneguzzi, « LSTM-Based Goal Recognition in Latent Space, » *CoRR*, vol. abs/1808.05249, 2018. Disponible à http://arxiv.org/abs/1808.05249

[2] M. Asai A. Fukunaga, « Classical Planning in Deep Latent Space : Bridging the Subsymbolic-Symbolic Boundary, » dans *AAAI 2018*, 2018. Disponible à https://aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16302

[3] D. Avrahami-Zilberbrand G. A. Kaminka, « Fast and Complete Symbolic Plan Recognition, » dans *IJCAI 2005*, 2005, pp. 653–658.

[4] C. L. Baker, R. Saxe, J. B. Tenenbaum, « Action understanding as inverse planning, » *Cognition 2009*, vol. 113, no. 3, pp. 329–349, 2009.

[5] F. Bisson, F. Kabanza, A. R. Benaskeur, H. Irandoust, « Provoking Opponents to Facilitate the Recognition of their Intentions, » dans *AAAI*, 2011.

[6] F. Bisson, H. Larochelle, F. Kabanza, « Using a Recursive Neural Network to Learn an Agent's Decision Model for Plan Recognition, » dans *IJCAI 2015*, 2015, pp. 918–924.

[7] F. Bre, J. Gimenez, V. Fachinotti, « Prediction of wind pressure coefficients on building surfaces using Artificial Neural Networks, » *Energy and Buildings*, vol. 158, 11 2017.

[8] H. H. Bui, S. Venkatesh, G. A. W. West, « Policy Recognition in the Abstract Hidden Markov Model, » *JAIR*, vol. 17, pp. 451–499, 2002.

[9] E. Charniak R. P. Goldman, « A Bayesian Model of Plan Recognition, » *Artif. Intell.*, vol. 64, no. 1, pp. 53–79, 1993. Disponible à https://doi.org/10.1016/0004-3702(93)90060-O

[10] C. Chen, X. Zhang, S. Ju, C. Fu, C. Tang, J. Zhou, X. Li, « AntProphet : an Intention Mining System behind Alipay's Intelligent Customer Service Bot, » dans *IJCAI 2019*, 08 2019, pp. 6497–6499.

[11] G. F. Cooper, « The Computational Complexity of Probabilistic Inference Using Bayesian Belief Networks (Research Note), » *Artificial Intelligence*, vol. 42, no. 2-3, pp. 393–405, mars 1990. Disponible à http://dx.doi.org/10.1016/0004-3702(90)90060-D

[12] A. Dosovitskiy V. Koltun, « Learning to Act by Predicting the Future, » dans *ICLR 2017*, 2017. Disponible à https://openreview.net/forum?id=rJLS7qKel

[13] Y. E.-Martín, M. D. R.-Moreno, D. E. Smith, « A Fast Goal Recognition Technique Based on Interaction Estimates, » dans *IJCAI 2015*, 2015, pp. 761–768.

[14] R. G. Freedman S. Zilberstein, « Integration of Planning with Recognition for Responsive Interaction Using Classical Planners, » dans *AAAI*, 2017.

[15] J. Fu, A. K. Balan, S. Levine, S. Guadarrama, « From Language to Goals : Inverse Reinforcement Learning for Vision-Based Instruction Following, » *ArXiv*, vol. abs/1902.07742, 2019.

[16] C. W. Geib R. P. Goldman, « Requirements for Plan Recognition in Network Security Systems, » dans *International Symposium on Recent Advances in Intrusion Detection 2002*, 2002.

[17] C. W. Geib R. P. Goldman, « A probabilistic plan recognition algorithm based on plan tree grammars, » *Artificial Intelligence*, vol. 173, no. 11, pp. 1101–1132, 2009.

[18] I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning*. MIT Press, 2016.

[19] R. Granada, R. Pereira, J. Monteiro, R. Barros, D. Ruiz, F. Meneguzzi, « Hybrid Activity and Plan Recognition for Video Streams, » dans *AAAI 2017*, 2017.

[20] D. Ha J. Schmidhuber, « Recurrent World Models Facilitate Policy Evolution, » dans *NeurIPS 2018*, 2018, pp. 2450–2462. Disponible à http://papers.nips.cc/paper/7512-recurrent-world-models-facilitate-policy-evolution.pdf

[21] D. Hana, Q. Liu, W. Fan, « A New Image Classification Method Using CNN transfer learning and Web Data Augmentation, » *Expert Systems with Applications*, vol. 95, 11 2017.

[22] K. He, X. Zhang, S. Ren, J. Sun, « Delving Deep into Rectifiers : Surpassing Human-Level Performance on ImageNet Classification, » *CoRR*, vol. abs/1502.01852, 2015. Disponible à http://arxiv.org/abs/1502.01852

[23] J. Hou, X. Wu, J. Chen, J. Luo, Y. Jia, « Unsupervised Deep Learning of Mid-Level Video Representation for Action Recognition, » dans *AAAI 2018*, 2018, pp. 6910–6917.

[24] Y. Jia, Y. Zhang, R. Weiss, Q. Wang, J. Shen, F. Ren, z. Chen, P. Nguyen, R. Pang, I. Lopez Moreno, Y. Wu, « Transfer Learning from Speaker Verification to Multispeaker Text-To-Speech Synthesis, » dans *Advances in Neural Information Processing Systems 31* S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett, éditeurs. Curran Associates, Inc., 2018, pp. 4480–4490. Disponible à http://papers.nips.cc/paper/7700-transfer-learning-from-speaker-verification-to-multispeaker-text-to-speech-synthesis.pdf

[25] F. Kabanza, J. Filion, A. R. Benaskeur, H. Irandoust, « Controlling the Hypothesis Space in Probabilistic Plan Recognition, » dans *IJCAI 2013*, 2013, pp. 2306–2312.

[26] H. A. Kautz J. F. Allen, « Generalized Plan Recognition, » dans *Proceedings of the Fifth AAAI National Conference on Artificial Intelligence*, série AAAI'86. AAAI Press, 1986, pp. 32–37. Disponible à http://dl.acm.org/citation.cfm?id=2887770.2887776

[27] N. R. Ke, A. Singh, A. Touati, A. Goyal, Y. Bengio, D. Parikh, D. Batra, « Modeling the Long Term Future in Model-Based Reinforcement Learning, » dans *ICLR 2019*, 2019. Disponible à https://openreview.net/forum?id=SkgQBn0cF7

[28] S. Keren, A. Gal, E. Karpas, « Goal Recognition Design with Non-Observable Actions, » dans *AAAI 2016*, 2016. Disponible à https://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12222/12074

[29] D. P. Kingma J. Ba, « Adam : A Method for Stochastic Optimization, » *CoRR*, vol. abs/1412.6980, 2014.

[30] Y. Kong Y. Fu, « Human Action Recognition and Prediction : A Survey, » *CoRR*, vol. abs/1806.11230, 2018. Disponible à http://arxiv.org/abs/1806.11230

[31] O. Köpüklü, M. Babaee, S. Hörmann, G. Rigoll, « Convolutional Neural Networks with Layer Reuse, » *CoRR*, vol. abs/1901.09615, 2019. Disponible à http://arxiv.org/abs/1901.09615

[32] A. Lerner, Y. Chrysanthou, D. Lischinski, « Crowds by Example, » *Comput. Graph. Forum*, vol. 26, pp. 655–664, 2007.

[33] K. Liu, W. Liu, C. Gan, M. Tan, H. Ma, « T-C3D : Temporal Convolutional 3D Network for Real-Time Action Recognition, » *AAAI*, 2018. Disponible à https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17205/16305

[34] P. Masters S. Sardiña, « Cost-Based Goal Recognition for Path-Planning, » dans *AAMAS 2017*, 2017, pp. 750–758.

[35] P. Masters S. Sardiña, « Cost-Based Goal Recognition in Navigational Domains, » *JAIR*, vol. 64, pp. 197–242, 2019. Disponible à https://doi.org/10.1613/jair.1.11343

[36] P. Masters S. Sardina, « Goal Recognition for Rational and Irrational Agents, » dans *AAMAS 2019*, Richland, SC, 2019, pp. 440–448. Disponible à http://dl.acm.org/citation.cfm?id=3306127.3331725

[37] W. Min, E. Ha, J. P. Rowe, B. W. Mott, J. C. Lester, « Deep Learning-Based Goal Recognition in Open-Ended Digital Games, » dans *AIIDE 2014*, 2014.

[38] W. Min, B. W. Mott, J. P. Rowe, B. Liu, J. C. Lester, « Player Goal Recognition in Open-World Digital Games with Long Short-Term Memory Networks, » dans *IJCAI 2016*, 2016, pp. 2590–2596.

[39] W. Min, B. Mott, J. Rowe, R. Taylor, E. Wiebe, K. Boyer, J. Lester, « Multimodal Goal Recognition in Open-World Digital Games, » dans *AAAI 2017*, 2017. Disponible à https://aaai.org/ocs/index.php/AIIDE/AIIDE17/paper/view/15910

[40] A. Y. Ng S. J. Russell, « Algorithms for Inverse Reinforcement Learning, » dans *Proceedings of the Seventeenth International Conference on Machine Learning*, série ICML '00. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 2000, pp. 663–670. Disponible à http://dl.acm.org/citation.cfm?id=645529.657801

[41] S. J. Pan, J. T. Kwok, Q. Yang, « Transfer Learning via Dimensionality Reduction, » dans *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2*, série AAAI'08. AAAI Press, 2008, pp. 677–682. Disponible à http://dl.acm.org/citation.cfm?id=1620163.1620177

[42] R. F. Pereira, N. Oren, F. Meneguzzi, « Landmark-Based Heuristics for Goal Recognition, » dans *AAAI 2017*, 2017, pp. 3622–3628.

[43] R. F. Pereira, M. Vered, F. Meneguzzi, M. Ramírez, « Online Probabilistic Goal Recognition over Nominal Models, » dans *IJCAI 2019*, 2019, pp. 5547–5553. Disponible à https://doi.org/10.24963/ijcai.2019/770

[44] S. Racanière, T. Weber, D. P. Reichert, L. Buesing, A. Guez, D. J. Rezende, A. P. Badia, O. Vinyals, N. Heess, Y. Li, R. Pascanu, P. W. Battaglia, D. Hassabis, D. Silver, D. Wierstra, « Imagination-Augmented Agents for Deep Reinforcement Learning, » dans *NIPS 2017*, 2017, pp. 5690–5701. Disponible à http://papers.nips.cc/paper/7152-imagination-augmented-agents-for-deep-reinforcement-learning

[45] M. Ramírez H. Geffner, « Plan Recognition as Planning, » dans *IJCAI 2009*, 2009, pp. 1778–1783.

[46] M. Ramírez H. Geffner, « Probabilistic Plan Recognition Using Off-the-Shelf Classical Planners, » dans *AAAI 2010*, 2010.

[47] S. Ravi H. Larochelle, « Optimization as a Model for Few-Shot Learning, » dans *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017. Disponible à https://openreview.net/forum?id=rJY0-Kcll

[48] N. Rhinehart K. M. Kitani, « First-Person Activity Forecasting with Online Inverse Reinforcement Learning, » *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 3716–3725, 2017.

[49] A. Sadilek H. A. Kautz, « Recognizing Multi-Agent Activities from GPS Data, » dans *AAAI 2010*, 2010. Disponible à http://www.aaai.org/ocs/index.php/AAAI/AAAI10/paper/view/1603

[50] C. F. Schmidt, N. S. Sridharan, J. L. Goodson, « The Plan Recognition Problem : An Intersection of Psychology and Artificial Intelligence, » *Artificial Intelligence*, vol. 11, no. 1-2, pp. 45–83, 1978.

[51] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, D. Hassabis, « Mastering the Game of Go with Deep Neural Networks and Tree Search, » *Nature*, vol. 529, no. 7587, pp. 484–489, January 2016.

[52] K. Simonyan A. Zisserman, « Two-Stream Convolutional Networks for Action Recognition in Videos, » dans *NIPS 2014* Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, K. Q. Weinberger, éditeurs, 2014, pp. 568–576. Disponible à http://papers.nips.cc/paper/5353-two-stream-convolutional-networks-for-action-recognition-in-videos.pdf

[53] S. Sohrabi, A. V. Riabov, O. Udrea, « Plan Recognition As Planning Revisited, » dans *IJCAI 2016*, série IJCAI'16. AAAI Press, 2016, pp. 3258–3264. Disponible à http://dl.acm.org/citation.cfm?id=3061053.3061077

[54] Y. C. Song, H. A. Kautz, J. F. Allen, M. D. Swift, Y. Li, J. Luo, C. Zhang, « A Markov logic framework for recognizing complex events from multimodal data, » dans *ICMI 2013*, 2013, pp. 141–148.

[55] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, « Dropout : a simple way to prevent neural networks from overfitting, » *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[56] N. Sturtevant, « Benchmarks for Grid-Based Pathfinding, » *Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 2, pp. 144 – 148, 2012. Disponible à http://web.cs.du.edu/~sturtevant/papers/benchmarks.pdf

[57] G. Sukthankar K. Sycara, « A cost minimization approach to human behavior recognition, » dans *AAMAS 2005*, 01 2005, pp. 1067–1074.

[58] G. Sukthankar, C. Geib, H. H. Bui, D. Pynadath, R. P. Goldman, *Plan, Activity, and Intent Recognition : Theory and Practice*, 1st édition. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 2014.

[59] A. Tamar, Y. Wu, G. Thomas, S. Levine, P. Abbeel, « Value Iteration Networks, » dans *IJCAI 2017*, 2017, pp. 4949–4953.

[60] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, C. Liu, « A Survey on Deep Transfer Learning, » dans *Artificial Neural Networks and Machine Learning – ICANN 2018* V. Kůrková, Y. Manolopoulos, B. Hammer, L. Iliadis, I. Maglogiannis, éditeurs. Cham : Springer International Publishing, 2018, pp. 270–279.

[61] M. Vered G. A. Kaminka, « Heuristic Online Goal Recognition in Continuous Domains, » dans *IJCAI 2017*, 2017, pp. 4447–4454.

[62] M. Vered, G. Kaminka, S. Biham, « Online goal recognition through mirroring : humans and agents, » dans *Fourth Annual Conference on Advances in Cognitive Systems*, 2016.

[63] M. Vered, R. F. Pereira, M. C. Magnaguagno, G. A. Kaminka, F. Meneguzzi, « Towards Online Goal Recognition Combining Goal Mirroring and Landmarks, » dans *AAMAS 2018*, 2018, pp. 2112–2114. Disponible à http://dl.acm.org/citation.cfm?id=3238089

[64] O. Vinyals, I. Babuschkin, J. Chung, M. Mathieu, M. Jaderberg, « AlphaStar : Mastering the Real-Time Strategy Game StarCraft II, » 2019. Disponible à https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/

[65] B. Volz, K. Behrendt, H. Mielenz, I. Gilitschenski, R. Siegwart, J. Nieto, « A data-driven approach for pedestrian intention estimation, » dans *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, Nov 2016, pp. 2607–2612.

[66] J. Wang, Y. Chen, S. Hao, X. Peng, H. Lisha, « Deep Learning for Sensor-based Activity Recognition : A Survey, » *Pattern Recognition Letters*, 07 2017.

[67] N. Watters, A. Tacchetti, T. Weber, R. Pascanu, P. Battaglia, D. Zoran, « Visual Interaction Networks, » *CoRR*, vol. abs/1706.01433, 2017.

[68] T. Wen, Y. Miao, P. Blunsom, S. J. Young, « Latent Intention Dialogue Models, » dans *ICML 2017*, 2017, pp. 3732–3741. Disponible à http://proceedings.mlr.press/v70/wen17a.html

[69] J. Wu, A. Osuntogun, T. Choudhury, M. Philipose, J. M. Rehg, « A Scalable Approach to Activity Recognition based on Object Use, » dans *ICCV 2007*, 2007.

[70] K. Xu, E. Ratner, A. Dragan, S. Levine, C. Finn, « Learning a Prior over Intent via Meta-Inverse Reinforcement Learning, » dans *Proceedings of the 36th International Conference on Machine Learning*, série Proceedings of Machine Learning Research K. Chaudhuri R. Salakhutdinov, éditeurs, vol. 97. Long Beach, California, USA : PMLR, 09–15 Jun 2019, pp. 6952–6962. Disponible à http://proceedings.mlr.press/v97/xu19d.html

[71] S. Yan, Y. Teng, J. S. Smith, B. Zhang, « Driver behavior recognition based on deep convolutional neural networks, » dans *ICNC-FSKD 2016*, 2016, pp. 636–641.

[72] S. Yan, Y. Xiong, D. Lin, « Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition, » *AAAI*, 2018. Disponible à https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17135/16343

[73] W. Zhao S. Du, « Spectral–Spatial Feature Extraction for Hyperspectral Image Classification : A Dimension Reduction and Deep Learning Approach, » *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 8, pp. 4544–4554, Aug 2016.