Analyse en-ligne de tracés manuscrits d'opérations arithmétiques posées pour l'aide à l'apprentissage sur tablette numérique

Arnaud Lods

COLLEGE MATHS, TELECOMS
DOCTORAL INFORMATIQUE, SIGNAL
BRETAGNE SYSTEMES, ELECTRONIQUE

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

THÈSE DE DOCTORAT DE

L'INSTITUT NATIONAL DES SCIENCES APPLIQUÉES DE RENNES

ÉCOLE DOCTORALE N° 601
Mathématiques, Télécommunications, Informatique, Signal, Systèmes, Électronique
Spécialité : *Informatique*

Par
Arnaud LODS

Analyse en-ligne de tracés manuscrits d'opérations arithmétiques posées pour l'aide à l'apprentissage sur tablette numérique

Thèse présentée et soutenue à Rennes, le 13 Septembre 2023
Unité de recherche : IRISA - UMR6074

Rapporteurs avant soutenance :

Jean-Yves RAMEL Professeur des Universités, Université de Tours
Armelle BRUN Professeur des Universités, Université de Lorraine

Composition du Jury :

Président :	Harold MOUCHÈRE	Professeur des Universités, Université de Nantes
Examinateurs :	Jean-Yves RAMEL	Professeur des Universités, Université de Tours
	Armelle BRUN	Professeur des Universités, Université de Lorraine
	Gwénolé LECORVÉ	Chercheur, Orange
Dir. de thèse :	Éric ANQUETIL	Professeur des Universités, INSA Rennes
Co-Encadrant de thèse :	Sébastien MACÉ	Responsable R&D, Learn&Go

Invité(s) :

Remerciements

Je souhaite tout d'abord remercier les membres du jury. Je remercie Jean-Yves Ramel, Professeur à l'Université de Tours, et Armelle Brun, Professeure à l'Université de Lorraine, d'avoir accepté de rapporter ce manuscrit. De même, je remercie Harold Mouchère, Professeur à l'Université de Nantes, et Gwénolé Lecorvé, Chercher à Orange d'avoir accepté d'être les examinateurs de ma soutenance.

Je tiens ensuite à remercier très chaleureusement Éric Anquetil, Professeur à l'INSA de Rennes, et Sébastien Macé, Responsable R&D à Learn&Go, qui m'ont donné l'opportunité de travailler sur ce sujet de thèse et m'ont accompagné par leur expertise dans la progression des travaux. Mais je souhaite aussi les remercier pour leur patience et leur accompagnement sur ces trop nombreuses années, ils m'ont permis de mener à bout ce travail malgré les difficultés que j'ai pu rencontrer.

Dans la continuité, je souhaite remercier à la fois les membres de l'équipe Shadoc dont les échanges en tout genre ont pu stimuler mes années de recherche, et les membres de l'équipe Learn&Go, grâce à qui j'ai pu valider les travaux présentés par la collaboration avec les écoles.

Je souhaite remercier mon entourage ; famille, collègue et amis. Même si je ne suis pas le plus expressif ou le plus présent, votre présence ont su me garder motivé tout au long du voyage.

Enfin, merci pour tout Pierre.

Table des matières

Liste des Figures

Liste des tableaux

Chapitre 1

Introduction

Dans ce chapitre nous introduisons les différents axes de recherche étudiés. L'objectif de nos travaux concerne l'aide à l'apprentissage des mathématiques par un accompagnement de la réalisation de pose d'opérations arithmétiques sur tablettes numériques accompagnées de stylets. En réalisant un état des lieux de l'enseignement des mathématiques en primaire, nous présentons l'intérêt d'un tel système. Nous serons amenés à résoudre plusieurs challenges scientifiques : la reconnaissance de structures 2D manuscrites et l'analyse de ces structures, afin de pouvoir produire des feedbacks personnalisés permettant de guider un élève dans la correction de ses opérations.

Sommaire

1.1 Mathématiques en primaire

"Pose l'opération 19786 + 215 + 3291". C'est un exercice proposé à des élèves de CM2 en 1987, 1999, 2007 et finalement 2017, qui vient conclure une étude de 30 ans en France sur l'évolution de la maîtrise des mathématiques au terme du cycle élémentaire. Alors qu'en 1987 94% des élèves répondaient correctement, en 2007 seulement 83% des élèves proposaient une réponse valide. Avec la multiplication "247 x 36", les taux de réussite baissent de 84% à 68%. Confrontés à des problèmes similaires, les élèves de primaire obtiennent de moins bons résultats. C'est un constat de plusieurs décennies puisque de nouveau observé en 2017 : le niveau scolaire moyen français en mathématiques régresse fortement. On retrouve ainsi des conclusions similaires dans les études TIMSS [1] ou Cedre [2]. Lorsque ces acquis ne sont pas assimilés, on constate que cette tendance s'observe aussi dans la suite des études (étude PISA [3] sur des élèves de 15 ans). Et bien au-delà : les récentes réformes du lycée rendant l'enseignement des mathématiques falcultatif pour les élèves ont participé à la régression du niveau en mathématiques des étudiants à l'université, qui n'ont pas le niveau nécessaire pour leur poursuite d'études sans phase de remédiation. À ce titre, un retour en arrière de la part du gouvernement français est opéré en 2023, souhaitant réinsérer l'enseignement des mathématiques obligatoire dans le cursus des lycéens, tout en appuyant d'avantage cet enseignement dès la primaire.

Une des observations expliquant la régression des résultats : les conditions de tutorat des élèves. Hors les études au niveau éducatif à tout niveau scolaire sont unanimes : un tutorat personnel apporte de meilleurs résultats pour la progression d'un élève. Mais il n'est pas possible pour un professeur de tutorer personnellement chaque élève dans une classe de 20 à 30 élèves, afin d'encadrer sa progression, de l'accompagner et de lui apporter des conseils personnalisés et immédiats durant son apprentissage. De plus, les difficultés de recrutement d'enseignants amènent les institutions à recruter des enseignants moins formés alors que les méthodologies d'enseignements continuent d'évoluer. Ainsi paradoxalement, alors que les outils évoluent, que l'accès aux ressources éducatives est de plus en plus simple, et que l'on évalue mieux le système éducatif, les enseignants et les parents se retrouvent de plus en plus en difficulté. Il devient complexe donc de s'attarder sur les difficultés rencontrées de chaque élève. Les délais ou l'absence de retours spécifiques

1. https ://www.education.gouv.fr/timss-2019-l-etude-internationale-consacree-aux-mathematiques-et-aux-sciences-11930
2. https ://www.education.gouv.fr/cedre-2008-2014-2019-mathematiques-en-fin-d-ecole-des-resultats-en-baisse-306336
3. https ://data.oecd.org/fr/pisa/competences-en-mathematiques-pisa.htm

pour un élève ne lui permettent pas de consolider ses bases, qui entraînent par la suite l'enchaînement observé d'échecs. Contraint par les effectifs limités du corps enseignant, comment réussir à apporter des retours personnalisés et immédiats à chaque élève ?

C'est la problématique que nous souhaitons traiter au cours de ces travaux de recherche. en proposant une évolution des outils mis à disposition des enseignants pour accompagner les élèves dans leur parcours scolaire. Il est difficile pour un enseignant d'attribuer le temps nécessaire à chaque élève pour identifier chacune des lacunes et les combler. Ainsi, être en mesure d'automatiser la détection des difficultés d'un élève devient un point central d'amélioration dans le parcours d'apprentissage. Pour les mathématiques, le champ du domaine est large. La résolution de problèmes, le calcul en ligne, le calcul mental, la géométrie ou la pose d'opération sont d'autant de sujets potentiels. Dans le cadre des travaux de cette thèse, nous traitons la problématique de la pose d'opérations. Nous priorisons la pose d'opérations car la résolution d'un tel problème correspond à une application algorithmique stricte de la part de l'élève. Ce champ restreint de solutions permet de réaliser des retours d'autant plus pertinents pour chaque réalisation. Ensuite, cela représente la base fondamentale de l'apprentissage des nombres. Comprendre les unités, les dizaines, comment elles interagissent entre elles pour des retenues... Consolider cette connaissance, c'est consolider les bases de l'enseignement mathématique d'un enfant. Ces travaux pourront, par la suite, ouvrir la porte à une étude sur d'autres typologies d'exercices rencontrés en école primaire.

On va ainsi concentrer notre discours sur les **feedbacks** que l'on souhaite mettre en place pour la pose d'opérations arithmétiques. Nous définissons un **feedback** comme étant un retour visuel fait à l'enfant sur sa saisie afin de lui signifier ses erreurs et de lui donner une correction ou des pistes l'amenant à corriger ses erreurs. La figure 1.1 représente l'exemple de trois opérations arithmétiques, une addition et deux soustractions, saisies par des élèves de primaire sur une tablette. On peut y voir des exemples typiques d'opérations que nous souhaitons pouvoir traiter afin de produire une analyse automatisée. Ces opérations contiennent des erreurs dans les chiffres du résultat : il manque une retenue au millier pour l'addition et le résultat des centaines pour une soustraction est éronné. De plus, on remarque des tracés au sein (ou autour) de l'opérations qui sont soit du bruit ou soit des tracés de guidage qu'a réalisé l'élève. Dans les deux cas, ces tracés ne doivent pas compromettre la compréhension de l'opération, puisqu'ils ne doivent pas être considérés comme incorrect.

Ce sont des tracés qui sont observés et répétés par de nombreux élèves, qui parfois ne

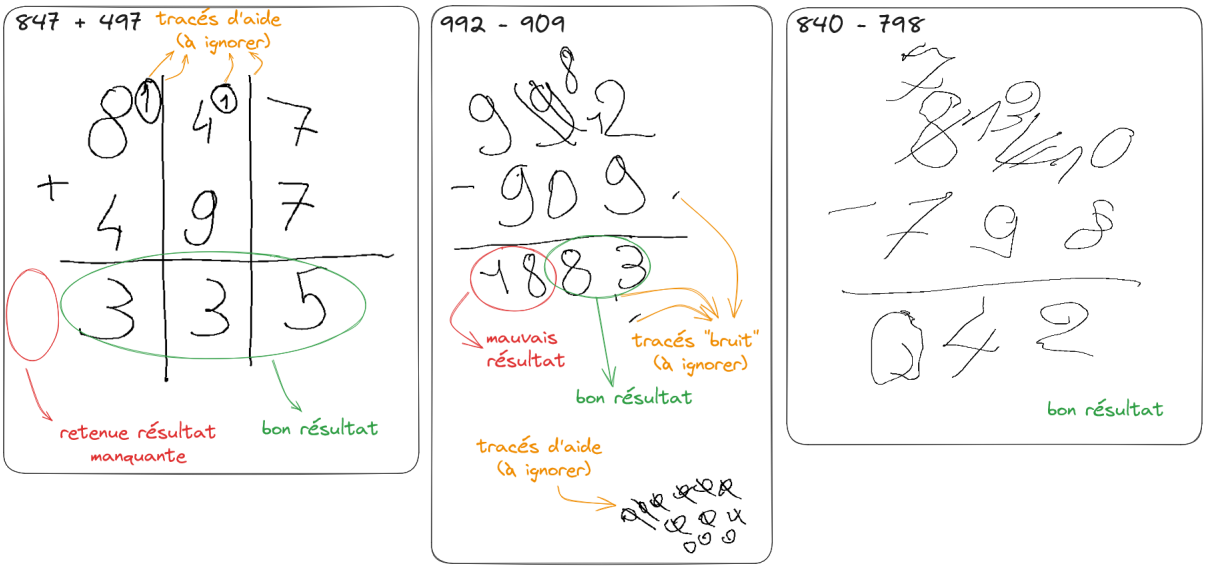

FIGURE 1.1 – Une addition et deux soustractions posées saisies par des élèves de primaire. On y constate deux typologies d'erreurs (mauvais chiffre, chiffre manquant), mais aussi des tracés de construction au sein de l'opération qui ne doivent pas compromettre l'interprétation de l'opération.

constituent pas des erreurs qui doivent être explicitement relevées, mais qui ne doivent pas non plus évidemment impacter les conclusions de l'analyse. On se retrouve donc avec de nombreux tracés que l'on va devoir être capable de discriminer, mais qui vont venir aussi perturber les informations extraites. Il ne faut pas que des tracés viennent empiéter sur des relations ou des positions estimées entre symboles, ou que des tracés comme des cercles et des ratures viennent aussi modifier la nature des données. Nous allons donc mettre en place des stratégies qui permettent de compenser ces erreurs.

Nous appellerons par la suite **feedbacks** les indications visuelles faites à l'élève dont nous souhaitons automatiser le traitement. Ces feedbacks ont pour but d'identifier et de représenter les conclusions que l'on peut porter sur la réalisation d'un élève : ses réussites, ses échecs et surtout la compréhension et la transmission des erreurs de l'élève. Ils peuvent être différents suivant des paramètres externes, tels que des difficultés précédemment rencontrées, son niveau d'apprentissage ou la nature de l'exercice. Il faut pouvoir les formaliser et les expliquer dans la continuité de la saisie de l'élève. Avec comme objectif : pouvoir faciliter le travail d'un enseignant en apportant des retours rapides à chaque élève sur ses erreurs. Il devient ainsi plus simple de déterminer un parcours d'apprentissage personnel pour surmonter les échecs. Les travaux de thèse se concentrent sur les

problématiques scientifiques associées à la production des feedbacks, à savoir la reconnaissance et l'analyse 2D manuscrites des opérations. La thématique de recherche n'étant pas centrée sur l'éducation, nous ne traiterons qu'un premier niveau de génération de feedbacks, identifiant et expliquant la nature des erreurs observées. Cette thèse CIFRE s'inscrit dans la continuité du Labcom ANR "Script&Labs" [4] qui a permis de concevoir la solution Kaligo présentée dans la section suivante.

1.2 Kaligo : le cahier d'apprentissage de l'écriture

C'est la collaboration de recherche entre l'équipe Shadoc du laboratoire IRISA et l'entreprise Learn&Go qui a donné naissance à ces travaux de recherche. **L'équipe Shadoc** (précédemment Intuidoc) du laboratoire IRISA apporte son savoir-faire sur la reconnaissance de documents manuscrits par l'interprétation de tracés en-ligne, développant un moteur d'analyse permettant les feedbacks à l'enfant. L'importance de l'écriture manuscrite sera développée dans la suite du manuscrit, et c'est l'analyse fine de cette écriture manuscrite en considérant le sens, la direction, la forme et la position des tracés qui est au coeur des sujets de recherche de l'équipe. **L'entreprise Learn&Go** coordonne les équipes du laboratoire, les écoles et les enseignants afin de définir les besoins, et d'intégrer les connaissances et moteurs d'intelligence artificielle conçus par les équipes de recherche pour la résolution des problématiques d'apprentissage. Elle offre l'environnement idéal afin d'avoir accès aux données nécessaires à l'apprentissage des modèles présentés, et les compétences adéquates à la mise en pratique de ces travaux.

Learn&Go s'engage sur l'aide à l'apprentissage à l'école au travers des nouveaux outils numériques centrés autour de l'écriture manuscrite. L'objectif est de profiter des progrès sur les interfaces tablettes stylets, afin de concevoir des systèmes intelligents permettant de guider les enfants dans la réalisation et la correction de leurs erreurs. On ne souhaite pas changer la méthode d'apprentissage, qui doit se faire à la main. Cette collaboration a mené à la conception de Kaligo, un outil d'analyse d'écriture manuscrite sur tablette permettant d'accompagner l'enfant dans son apprentissage de l'écriture. L'outil permet, au travers de nombreux modèles d'écritures, de proposer à l'enfant des retours visuels sur la qualité de son écriture afin de l'aider à progresser et corriger ses erreurs. Un enseignant n'a plus besoin d'analyser des centaines de lettres réalisées par chaque enfant afin de lui rapporter ses erreurs, et chaque enfant peut immédiatement recevoir un feedback concernant son

4. http ://scriptandlabs.irisa.fr/

écriture, pour s'améliorer dans la continuité de l'exercice. La figure 1.2 présente le cas d'utilisation pour l'écriture d'un mot en lettres bâton majuscules. La solution Kaligo offre entre autres différents exercices permettant à l'enfant d'appréhender l'ensemble de l'écriture manuscrite, en passant par des exercices sur la pression du stylet sur la tablette, l'écriture cursive ou en bâton de lettres isolées ou de mots complets, mais peut être aussi étendue au travail de l'orthographe.

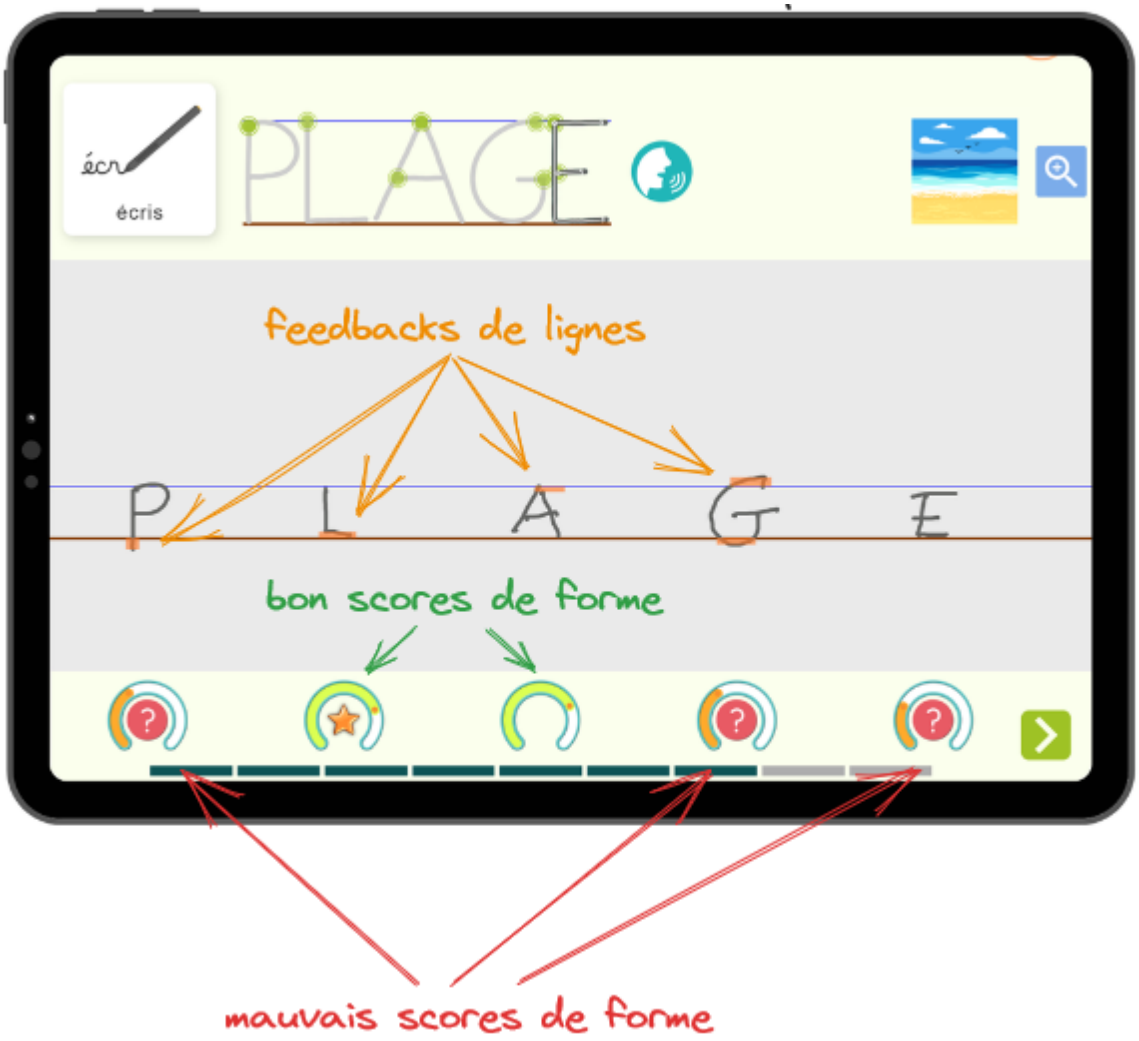

FIGURE 1.2 – L'application Kaligo, le cahier d'apprentissage de l'écriture. L'enfant obtient des retours sur chacune de ses lettres, sur le score de chaque lettre et avec des feedbacks personnalisés.

Ce projet pluridisciplinaire permet ainsi la collaboration de plusieurs unités de recherche dont le domaine de compétence central restera l'intelligence artificielle. Nous y

reviendrons, mais la capacité d'adaptation du système aux erreurs de l'élève est primordiale dans la réalisation d'une application pour la création de feedbacks pertinents. Il ne s'agit pas simplement d'avertir l'élève qu'une réalisation est incorrecte, mais bien de lui apporter des retours personnalisés pour ses erreurs, en lui donnant des pistes de résolution. Pouvoir comprendre et expliquer les erreurs de l'élève est au coeur des problématiques.

1.3 Évolution de Kaligo pour les mathématiques : axes de recherche de la thèse

Afin d'enrichir la solution Kaligo, le souhait est donc de permettre l'accompagnement des élèves dans l'apprentissage des mathématiques. L'objectif est de pouvoir offrir un outil intuitif basé sur l'utilisation de tablettes avec stylet simulant l'approche papier/crayon, dans un environnement numérique sans contraintes de réalisation, afin de rendre le transfert de connaissances entre supports numériques et papiers implicite. La thématique de la saisie manuscrite reste ainsi au coeur du projet. L'intérêt d'une telle solution est de pouvoir par la suite proposer des ensembles d'exercices se concentrant sur les difficultés rencontrées par chaque élève, basés sur ses propres réussites ou erreurs. Il sera alors possible de lui fournir un parcours pédagogique adéquat pour combler ses lacunes. L'objectif de cette thèse est de concevoir le système d'e-apprentissage pour l'acquisition du calcul arithmétique posé. La conception de ce système est caractérisé dans les grandes lignes par :

- La capacité de reconnaître rapidement les tracés manuscrits réalisés par un élève à l'aide d'un stylet sur tablette numérique pour la composition d'opérations arithmétiques posées. Il est important de pouvoir transmettre immédiatement des feedbacks à l'élève une fois son opération saisie pour que ses erreurs ne soient pas reproduites dans de multiples exercices avant d'obtenir un premier feedback.
- La possibilité d'interpréter la solution de l'enfant afin de comprendre les erreurs produites pour proposer des retours visuels adaptés directement sur sa production, en fonction des typologies d'erreurs, de la consigne et de sa progression au sein de l'application. En effet, il ne s'agit pas simplement de donner une évaluation d'un exercice, mais bien d'en expliquer les erreurs.

Plusieurs axes de recherche se dégagent de ces problématiques. Tout d'abord l'importance de la pertinence des feedbacks retournés à l'élève pour expliquer les erreurs observées. Pour produire ces feedbacks, il faut pouvoir mettre en parallèle une consigne,

et donc un résultat attendu, avec la réalisation de l'élève. C'est ce que l'on nommera la **mise en correspondance** entre un résultat attendu et la réalisation d'un élève et qui sera un de nos axes principaux de travail. Afin de mettre en lien une solution attendue avec un tracé manuscrit, il est nécessaire de pouvoir modéliser la réalisation de l'enfant pour l'interpréter. Ainsi, l'autre axe principal concerne **la reconnaissance des tracés manuscrits**. La **structure bi-dimensionnelle** d'opérations arithmétiques posées entraîne de nombreux challenges scientifiques autour de la reconnaissance et de l'interprétation de ces tracés manuscrits dans le contexte d'une opération structurée afin d'en déduire la nature des erreurs commises.

1.3.1 Premier axe de recherche : pédagogie du feedback immédiat

Afin que les feedbacks faits à l'élève soient pertinents, ils se doivent d'être personnalisés et immédiats pour s'inscrire dans un processus d'apprentissage l'amenant à corriger au plus tôt ses erreurs. Les contributions mises en avant doivent donc amener le système à proposer ces feedbacks directement à la suite de sa proposition. Un exemple sommaire de tels feedbacks sont présentés sur la figure 1.1. Ces feedbacks peuvent être de multiples natures en fonction de l'erreur observée : l'oubli d'une retenue, un mauvais résultat, un mauvais alignement des chiffres, une mauvaise recopie... Tous ces éléments doivent être déduits de l'interprétation réalisée sur l'ensemble de l'opération arithmétique et mettre en exergue les erreurs commises, en les identifiant correctement. Même si l'opération est incomplète, le système doit pouvoir être utilisé afin d'obtenir une compréhension des éléments manquants, différents ou inattendus dans l'opération. Un ensemble de règles peuvent ensuite être appliquées suivant le contexte pour générer un feedback adapté.

— La bonne composition de l'opération : aligner les chiffres dans les colonnes correspondantes, placer l'opérateur correctement, placer la plus grande opérande au dessus lors d'une soustraction ou encore placer les retenues au bon endroit ;
— Appliquer l'algorithme prédéfini : penser à l'utilisation de retenues, réaliser les opérations dans l'ordre ;
— Le résultat obtenu : que ce dernier corresponde au résultat attendu de l'opération.

Le système étant dédié à l'apprentissage de l'arithmétique à l'école primaire, il doit être intuitif à utiliser, adapté aux besoins des élèves et des enseignants. Dans la suite du manuscrit, nous nous concentrerons sur la nature des feedbacks fournis à l'élève. La mise

en forme de ces derniers et leur évolution dans un parcours d'apprentissage complet ne seront pas étudiés ici. Nous nous concentrerons sur la robustesse de ces feedbacks, à partir de la reconnaissance et de l'interprétation de la production manuscrite de l'élève et de leur utilisation immédiate dans le contexte de résolution d'un exercice.

1.3.2 Deuxième axe de recherche : reconnaissance des tracés manuscrits

Le deuxième axe des travaux porte sur la reconnaissance à la volée des tracés manuscrits de l'enfant. La production manuscrite est un ensemble de tracés représentant l'écriture de l'opération, c'est à dire l'écriture posée des éléments de la consigne correctement alignés, la présence d'opérateurs, de barre de séparation, puis l'application d'un algorithme de résolution connu avec la présence d'un résultat et de potentiels reports, retenues ou opérations transitoires dans le cadre de la multiplication et de la division. L'objectif est de permettre à l'enfant de réaliser sa saisie sans contraintes physiques, afin de se positionner dans une condition traditionnelle d'écriture d'opération similaire à celle sur papier. La plupart des outils existants aujourd'hui pour accompagner l'apprentissage de la réalisation d'opérations arithmétiques posées proposent des interfaces sur ordinateur utilisant un clavier, avec soit des espaces de saisie prédéfinis contraignants, soit des boîtes englobantes limitant et guidant l'élève, rendant moins efficace le transfert de l'apprentissage vers une saisie papier sans aide visuelle. Nous nous intéressons donc à une utilisation plus naturelle de la résolution d'opérations arithmétiques posées, afin de se rapprocher de celle réalisée sur un support papier. Pour cela, d'une part, il faut **reconnaître et segmenter** en symboles mathématiques les tracés. D'autre part, il faut identifier les relations spatiales entre symboles afin de composer la structure bi-dimensionnelle de l'opération. Le tout doit être réalisé avec la contrainte temporelle afin de pouvoir fournir des retours visuels à l'élève directement à la suite de sa proposition.

Nous proposons une contribution principale sur la représentation des opérations arithmétiques posées et sur leur structure en étendant et qualifiant la définition des graphes afin d'y porter une notion forte de visibilité. L'application de **graphes de visibilité** permettant de relier toutes paires de tracés pouvant avoir une signification structurelle, avec l'utilisation de **paysages de visibilité floue** pour définir les relations entre paires de tracés et paires de symboles. Cette représentation et ces caractéristiques permettent de modéliser entièrement les opérations arithmétiques sans perdre d'informations struc-

turelles en conservant toutes relations pertinentes entre symboles, et en proposant des caractéristiques adéquates pour représenter le positionnement des différents symboles mathématiques au sein de la structure. Afin d'éviter de produire des erreurs de reconnaissance dues à la grande variabilité de l'écriture manuscrite des élèves, nous proposons de générer et confronter de multiples hypothèses de reconnaissance, afin de ne pas introduire de potentielles erreurs de reconnaissance qui auraient un impact sur l'interprétation des tracés. Ces hypothèses seront traitées dans notre troisième axe de recherche.

1.3.3 Troisième axe de recherche : mise en correspondance entre consigne et réalisation

Le troisième axe des travaux porte sur la mise en correspondance de la solution de l'élève avec le résultat attendu. Suite aux premières contributions présentées, nous disposons d'une représentation de l'opération arithmétique posée sous la forme d'un graphe qui conserve la structure bi-dimensionnelle de l'opération. Cependant l'interprétation des tracés nécessite une compréhension de l'opération en rapport avec une solution attendue. Il est en effet nécessaire de trouver des correspondances entre la production de l'élève qui peut contenir plusieurs erreurs de différentes natures (voir figure 1.1) et la solution qui était attendue pour pouvoir interpréter sa réponse, afin d'identifier ses erreurs. L'objectif de ce troisième axe est donc de pouvoir confronter les tracés de l'enfant, qui peuvent ne pas respecter la structure attendue et contenir des erreurs, avec une ou plusieurs solutions idéales attendues. Il est aussi nécessaire de pouvoir réaliser cette interprétation rapidement, dans la continuité de sa proposition, afin de ne pas couper l'élève dans la réalisation d'exercices subséquents, en intervenant idéalement lorsque ce dernier est encore concentré sur cet exercice.

Nous apportons plusieurs contributions dans le cadre de cet axe de recherche. Premièrement, nous proposons l'application d'algorithmes sur les graphes pour générer des **mises en correspondance** entre paire de graphes. Pour ce faire, nous utilisons la représentation sous forme de graphes précédemment mentionnée et nous générons des modèles attendus correspondants, en utilisant des caractéristiques extraites pour identifier la meilleure correspondance. Nous proposons aussi d'utiliser cette mise en correspondance afin de sélectionner l'hypothèse la plus pertinente et identifier les tracés de l'élève. Pour contourner les problèmes de temporalité qui peuvent devenir exponentiels sur ces calculs, et ainsi produire une interprétation quasi temps réel, nous proposons dans un deuxième temps

plusieurs optimisations pour accélérer cet appariement. Nous présentons une stratégie de mise en correspondance partielle, et de nouvelles segmentations de graphes permettant de conserver la structure bi-dimensionnelle des opérations tout en réduisant l'espace de recherche, à la fois pour la création d'hypothèses et pour l'appariement en lui-même. Ces contributions permettent de produire une interprétation automatisée d'opérations en un temps satisfaisant, tout en identifiant de façon robuste les différences recherchées.

1.4 Organisation du manuscrit

Le chapitre 2 présente l'état de l'art relatif aux différentes problématiques scientifiques soulevées par ces travaux. Nous traitons d'abord des fondements pédagogiques de la thèse par le biais de tuteurs intelligents, leur importance dans l'accompagnement de l'apprentissage et les problématiques sous-jacentes à ces systèmes. La reconnaissance des tracés manuscrits dans le cadre d'opérations arithmétiques posées en 2D est ensuite abordée. Nous étudions différentes techniques et approches, et mettons en exergue chaque solution par rapport aux besoins globaux concernant l'interprétation des opérations, ainsi que les contraintes techniques survenant dans le contexte d'une interaction quasi temps-réel du système avec l'élève. Nous développons enfin les pistes nous permettant de pouvoir concevoir la solution d'interprétation des opérations arithmétiques afin de mettre en correspondance la reconnaissance des tracés de l'élève et la solution attendue déduite de la consigne de l'exercice.

Le chapitre 3 présente l'ensemble des travaux réalisés au cours de cette thèse. Nous commençons par présenter une vue d'ensemble du système proposé, afin de positionner nos différentes contributions dans la conception globale du système d'interprétation d'opérations arithmétiques. Nous traitons d'abord de nos contributions sur la reconnaissance à la volée des tracés manuscrits. Nous introduisons les modules de segmentation de tracés, de reconnaissance de symboles et de reconnaissance de relations entre symboles, afin de constituer des graphes représentant les opérations arithmétiques posées et leur structure bi-dimensionnelle. Nous détaillons ensuite l'interprétation des opérations arithmétiques. Nous y présentons notre module de mise en correspondance sur lequel se base la production des feedbacks, le principe général de l'approche et son interaction avec le module de reconnaissance pour la génération et la sélection d'hypothèses. Nous proposons une mise en correspondance partielle afin d'accélérer le processus, ainsi qu'une représentation intermédiaire des opérations arithmétiques permettant des traitements efficaces et rapides

pour répondre à la contrainte temps réel présentée. Enfin nous traitons de la génération de feedbacks sommaires à partir de ces mise en correspondances.

Le chapitre 4 présente les expérimentations de l'ensemble des travaux de thèse. Nous présentons le jeu de données et le protocole expérimental appliqué. Nous détaillons ensuite les performances des différents modules à la fois au niveau de la reconnaissance, de la mise en correspondance et de la qualité de l'interprétation des saisies des élèves par notre système. Enfin, un premier prototype développé sur tablette du système proposé est présenté.

Le chapitre 5 apporte une conclusion sur ces travaux de thèses et les perspectives liées à l'évolution de l'ensemble du système, à la fois pour des opérations plus complexes mais aussi sur d'autres bases de connaissances mathématiques. Nous ouvrons le sujet sur la richesse des feedbacks pouvant être générés ainsi que sur de futurs travaux étendant le système proposé à la problématique voisine de résolution d'opérations arithmétiques en ligne.

Chapitre 2

Etat de l'art

Ce chapitre traite de l'état de l'art des axes de recherches de la thèse. Nous commençons par présenter les fondements pédagogiques du numérique pour l'enseignement des mathématiques, et les contraintes associées. Nous nous intéressons en particulier à l'intérêt et aux difficultés d'y intégrer la saisie manuscrite. Cela permet d'introduire les problématiques scientifiques qui nous permettront de développer une analyse des opérations arithmétiques manuscrites : la reconnaissance d'opérations arithmétiques manuscrites et la mise en correspondance.

Sommaire

2.1 Fondements pédagogiques et scientifiques

Nous commençons par présenter les intérêts pédagogiques de l'introduction du numérique dans l'apprentissage des mathématiques. Nous le rappelons, la thèse n'est pas axée sur la pédagogie, nous nous concentrons sur les problématiques scientifiques liées à l'analyse des opérations arithmétiques posées. Cependant ces travaux ayant pour objectif de s'insérer dans un processus pédagogique, il est important d'étudier cet environnement. L'objectif est de présenter l'importance capitale de la saisie manuscrite dans le processus d'apprentissage et de restitution des compétences, et des contraintes liées à une telle saisie. Au travers de la présentation de *Systèmes Tutoriels Intelligents* (STI) existants, nous observons ce que l'on peut attendre d'un système dédié à l'accompagnement de l'apprentissage des mathématiques. Nous en déduisons les challenges scientifiques d'intelligence artificielle liés au traitement de la saisie manuscrite des opérations arithmétiques posées.

2.1.1 Numérique et écriture manuscrite

Le numérique a été introduit dans l'enseignement il y a plusieurs décennies avec deux objectifs principaux. Il était souhaitable d'introduire l'utilisation d'un ordinateur à l'école pour étendre l'apprentissage scolaire à ce nouvel outil et de se servir de cet outil pour développer et améliorer les apprentissages standards. Il existe aujourd'hui de très nombreuses applications visant à accompagner l'apprentissage des élèves en mathématiques, de l'école primaire au secondaire. Cet environnement permet d'enrichir l'expérience de l'élève en facilitant la visualisation de certains concepts, et en permettant de les manipuler différemment. De nombreux types d'exercices et formats peuvent être envisagés : les applications peuvent porter sur des calculs, des tables de multiplications, de l'arithmétique ou encore de la géométrie. On peut citer les travaux de Li [LM10] qui référencent un ensemble de 46 études concernant à la fois l'enseignement primaire et secondaire des mathématiques, et des impacts de ces outils numériques sur la progression de l'apprentissage chez les élèves. La plupart de ces études démontrent l'impact positif qu'a l'utilisation de tels logiciels sur les résultats des élèves. Ces évaluations sont réalisées en comparant les élèves disposant de ces outils avec des élèves utilisant les méthodes d'enseignements traditionnelles (dits groupes contrôles).

L'outil numérique permet d'accompagner l'apprentissage des mathématiques, cependant l'utilisation de ce dernier limite souvent la transférabilité des compétences apprises. Il est en effet important que l'apprentissage traditionnel sur papier puisse être facile-

ment appliqué à la solution numérique. Mais il est aussi important que toute approche numérique permette à l'élève de reproduire par la suite ses enseignements sur papier. Si l'approche permet d'appréhender les mathématiques, la résolution des exercices peut toutefois se retrouver impactée par la limitation des interfaces mise à la disposition des élèves.

Dans [AYK07], les auteurs s'intéressent ainsi à l'importance de l'écriture manuscrite dans l'apprentissage des mathématiques, et son impact sur le transfert entre le numérique et le papier. Ce concept est étendu dans [Ant+08] et dénote notamment les charges cognitives supplémentaires nécessaires à la maîtrise d'un nouvel outil, mais aussi de la satisfaction des élèves de pouvoir manipuler les outils par l'écriture. On constate que des interfaces webs communes basées sur des saisies claviers tel que IXL [1] (figure 2.1a) sont moins adaptées. D'autres outils reproduisent l'application de l'écriture manuscrite tel que MyBlee [2] (figure 2.1b), tout en restreignant l'espace de saisie pour l'élève. On parle de "saisie pré-casée", c'est un guidage pour l'élève qui constitue une aide mais aussi une contrainte puisqu'il doit respecter l'espace de saisie offert. Cela permet de faciliter l'analyse des productions manuscrites, puisqu'avec ce pré-casage des données saisies, les différents groupes de tracés sont préalablement séparées par symbole mathématique et alignés. Chaque symbole correspond donc à une position identifiée au sein de l'opération. L'élève se retrouve guidé implicitement dans les alignements proposés, à glisser ou au mieux écrire des retenues dans des cases prédéfinies. Ces éléments peuvent réduire la capacité de transfert des apprentissages de la pose d'opération entre la tablette et le papier, en ajoutant des contraintes et des supports dont l'élève ne disposera pas sur une feuille de papier. L'agencement des chiffres ou des opérateurs et l'application correcte de l'algorithme, des reports et des retenues, est aussi important lors du processus d'apprentissage de l'élève que l'obtention du bon résultat. Ce mode de saisie pré-casé est ici privilégié afin de contourner les limitations des capacités d'analyse de l'IA au sein d'une opération arithmétique représentée dans un espace en deux dimensions, composée de nombreux tracés manuscrits.

Nous cherchons dans cette thèse à bénéficier du numérique avec le support de l'IA, tout en restant au plus proche de l'approche traditionnelle papier/crayon de pose d'opérations que les élèves apprennent, sans restriction dans leur écriture. L'objectif principal est de s'émanciper de cette contrainte de saisie pré-casée en résolvant les différentes pro-

1. https :fr.ixl.commaths
2. https ://www.mybleemath.com/

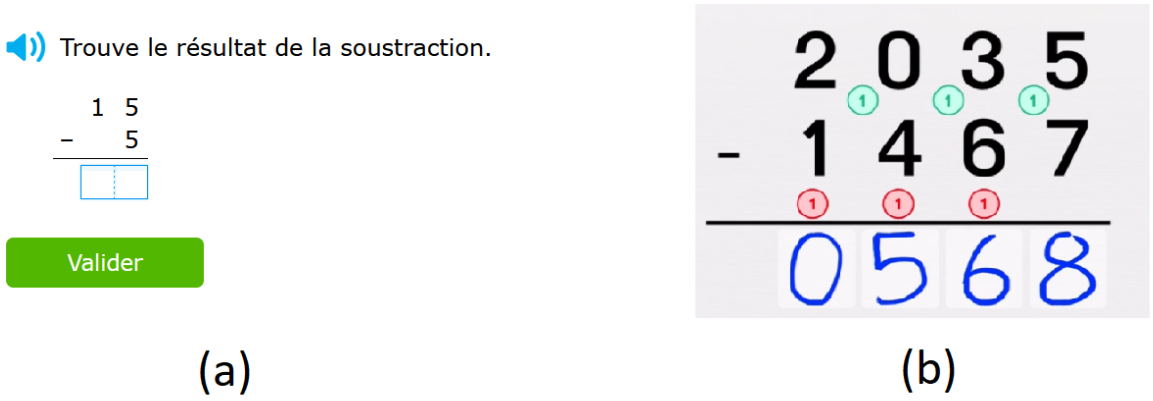

FIGURE 2.1 – Deux exemples d'applications proposant la pose "pré-casée" d'opérations arithmétiques. (a) IXL propose une interface web avec une saisie limitée au clavier. (b) MyBlee propose une application mobile utilisant l'écriture manuscrite mais avec des restrictions sur les positions et alignements de chiffres, d'opérateurs ou de reports.

blématiques liées à l'analyse d'opérations arithmétiques. Nous souhaitons proposer une interface la plus sobre possible pour l'élève, en permettant des saisies entièrement libres de ses opérations. Ainsi, un élève qui apprend et arrive à réaliser des opérations sur un tel outil numérique sera en capacité de les réaliser librement sans l'outil par la suite. Cela permet de garantir à la fois la transférabilité de l'apprentissage numérique sur une réalisation papier, de ne pas surcharger cognitivement l'élève et d'étendre l'apprentissage déjà réalisé sur papier sur les tablettes numériques. Il sera donc nécessaire de concevoir un système basé sur une IA capable de pouvoir interpréter les tracés manuscrits libres de l'enfant.

2.1.2 Feedbacks

Définition 1 (Feedback)**.** Un feedback est une observation délivrée à l'élève sur un travail réalisé, visant à mettre en exergue les réussites, les erreurs ou les recommandations étant donné une solution produite par un élève et une solution attendue par rapport à une consigne. Cela permet de guider l'élève dans son apprentissage.

Des feedbacks peuvent être différés ou immédiats. Le feedback différé peut s'apparenter au retour de l'enseignant sur la copie d'un élève. Il intervient après la réalisation de l'exercice, parfois plusieurs jours après. À l'inverse, le feedback immédiat peut être donné au cours de la réalisation de l'élève ou immédiatement à la suite de celle-ci. L'élève peut ainsi

avoir des retours avant la réalisation d'autres exercices et éviter ainsi une reproduction systématique de ses erreurs avant qu'une correction ne lui soit apportée.

De nombreuses études ont démontré l'intérêt des feedbacks immédiats par contraste aux feedbacks différés [Dih+04], et confirment l'impact bénéfique de la présence de ces feedbacks au cours de l'apprentissage. Il est cependant impossible pour un enseignant de fournir à chaque élève personnellement des retours immédiats sur sa production lors de la réalisation d'exercices. C'est ici tout l'intérêt d'une solution numérique basée sur une IA, qui permet de suivre l'apprentissage de chaque élève et de lui apporter des feedbacks personnalisés immédiats afin d'accompagner les réussites et échecs.

Parmi les typologies de feedbacks qui peuvent être réalisées, on en distingue deux principales : feedback "correctif", qui consiste à décrire les erreurs de l'élève, et un feedback "de guidage", qui intervient au cours de la pose de l'opération pour corriger au plus tôt de potentielles erreurs ou pour guider la prochaine étape de l'algorithme à appliquer sur l'opération. Pour pouvoir proposer ces deux typologies de feedbacks, il est nécessaire de coupler la reconnaissance des tracés manuscrits de l'enfant à une interprétation "sémantique" automatisée de ces tracés relativement à la consigne posée. Notre but est d'accompagner la conception de ce que l'on appelle un tuteur intelligent, qui permet la supervision de l'opération en temps réel, afin d'aider à l'apprentissage des calculs arithmétiques posés. À cette fin, il est nécessaire d'apporter à l'élève des feedbacks immédiats nécessaires à la bonne résolution de ses opérations afin d'éviter une propagation de ses erreurs. Le tuteur intelligent est constitué de trois phases permettant le traitement des saisies des élèves : une étape de reconnaissance, une étape d'interprétation et une étape définissant les feedbacks correctifs ou de guidage. Nous allons détailler par la suite le principe d'un tel système.

2.1.3 Système tutoriel intelligent

Définition 2 (Système tutoriel intelligent)**.** Un Système Tutoriel Intelligent (STI) est un système qui a pour objectif de fournir une instruction et des feedbacks, souvent immédiats, personnalisés pour chaque apprenant, sans intervention d'un enseignant humain. L'objectif est d'une part, de faciliter l'apprentissage de manière efficace grâce aux outils numériques, et d'autre part, de fournir des parcours de connaissances adaptés en fonction du niveau de chaque élève. Cela se traduit par la production de feedbacks individualisés et la capacité à concentrer l'apprenant sur des difficultés personnelles, chose qu'il est compliqué de reproduire sur un schéma typique avec un enseignant dans une classe où chaque élève

progresse à son rythme. Cela permet de maintenir un engagement constant de l'enfant sur son propre parcours de compétences, d'évoluer à son rythme et d'éviter la frustration d'introduire de nouveaux concepts quand les précédents ne sont encore maîtrisés.

Les auteurs dans [NMB10] notent une prolifération de tels systèmes dans les récentes années, avec la progression des supports numériques mais aussi des études notant l'impact sur l'apprentissage des feedbacks et parcours personnalisés au cours de l'apprentissage. Même si les architectures varient entre STI, une architecture générique est présentée dans la figure 2.2. Elle est composée de **quatre composants** communs : le **module du domaine**, le **module de l'apprenant**, le **module du tutorat**, et **l'interface**. Le **module du domaine** représente la connaissance du domaine d'application. Pour la pose d'opérations arithmétiques, elle est représentée par un ensemble de règles algorithmiques permettant de construire la pose et la résolution d'une telle opération. Le **module de l'apprenant** observe les actions de l'élève et construit une représentation de l'état de résolution. L'idée est de pouvoir ajuster le feedback en fonction des informations propres à l'élève, qui sont représentées soit par l'état de progression de la résolution de l'opération, ou soit par des connaissances plus précises telles que la maîtrise par l'élève de concepts de résolution. Le **module de tutorat**, en lien avec le module de l'apprenant, permet d'établir la stratégie d'enseignement, c'est-à-dire la génération de feedbacks ou d'indices afin d'interagir avec l'élève. Certains travaux [Py01] fusionnent le module du domaine et le module de tutorat pour créer un module expert complet. Enfin **l'interface** est le module de communication et d'interaction avec l'élève. L'Interface Homme-Machine (IHM) peut prendre plusieurs formes, et proposer différentes interactions (clavier, stylet, tactile) et actions (glisser/déposer, écrire, tracer...). Ce module ne doit pas être un obstacle à l'apprentissage, d'où notre proposition d'introduire une interface utilisant des tablettes avec stylet afin de rendre transparent l'utilisation et la manipulation d'un tel système. Cependant, la mise en place d'une telle interface complexifie la conception du module de l'apprenant, puisque les actions de l'élève doivent être interprétées à partir de la saisie manuscrite brute de l'élève. C'est l'objectif de ces travaux de thèse : concevoir une IA au coeur du module de l'apprenant permettant de réaliser cette interprétation des tracés manuscrits pour produire les feedbacks. Il est nécessaire de pouvoir reconnaître l'écriture manuscrite et le positionnement relatifs de tous les éléments de l'opération afin de déterminer de multiples typologies d'erreurs comme des erreurs de calculs, d'alignements ou d'oubli de retenues.

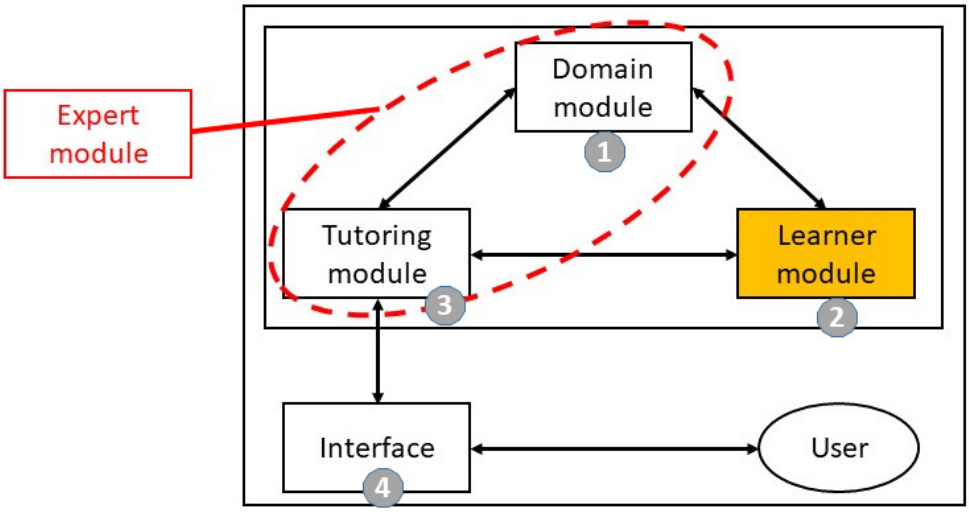

FIGURE 2.2 – L'architecture en 4 composants, (1) le module du domaine, (2) le module de l'apprenant, (3) le module de tutorat, (4) l'interface. [NMB10]

STI pour l'apprentissage des mathématiques.

Les auteurs [SC13] référencent un ensemble de 27 études entre 1997 et 2010 sur l'utilisation de STI pour l'apprentissage des mathématiques chez des élèves de primaire, collège et lycée. Les objectifs de ces études étaient multiples : la confrontation de l'utilisation de STI à l'enseignement traditionnel, le couplage de l'enseignement traditionnel avec l'utilisation d'un STI, ou encore le remplacement de travaux écrits à la maison par des travaux utilisant ces STI. Les conclusions de cette étude déterminent que si l'utilisation seule de STI n'apporte pas une meilleure réussite scolaire par rapport à un tutorat en face-à-face avec un enseignant humain, on trouve un intérêt à coupler un STI à l'enseignement traditionnel. Les élèves disposant à la fois de feedbacks réalisés par les STI et par l'enseignement standard d'un enseignant obtiennent de meilleurs résultats. Les observations tendent à confirmer que ces meilleures performances sont observées à la fois sur des périodes courtes (semaines), ou plus longues (mois / semestres) sur les évaluations des groupes expérimentaux par rapport aux groupes contrôles.

Les travaux de [GML08] présentent l'étude de l'application du STI eFit (figure 2.3) qui fût utilisée dans plusieurs écoles en Allemagne pour l'apprentissage des mathématiques. Les élèves sont testés sur diverses opérations arithmétiques (addition, soustraction, multiplication et division) tantôt simples, tantôt complexes. Le système conçoit ensuite un parcours spécialisé pour chaque enfant en fonction des scores obtenus sur les tests initiaux

FIGURE 2.3 – Capture de l'interface WEB de eFit [GML08], où la saisie des opérations se fait par une saisie clavier.

évaluant les lacunes de chaque élève. L'étude s'est poursuivie sur une année scolaire. 2 heures de cours de mathématiques par semaine étaient remplacées par 2 heures utilisant le système eFit pour le groupe expérimental. Les conclusions mettent en avant des performances notablement accrues sur les concepts mathématiques évalués au cours de l'année scolaire pour le groupe expérimental utilisant eFit. Dans [ZJ17], les auteurs évaluent Lexue 100, un autre STI pour l'apprentissage des mathématiques. De la même manière, le système analyse les réussites propres à chaque élève afin de leur apporter un parcours personnalisé de résolution adapté à ses difficultés. Si les groupes expérimentaux ont obtenu de meilleurs scores, il est à noter que l'évaluation psychologique des élèves ayant utilisé le système dénote souvent l'interface comme étant un obstacle à l'apprentissage, due à des interfaces impliquant une surcharge cognitive. Elles étaient soit non adaptées aux exercices, soit difficiles à maîtriser.

Les auteurs de [Ant+08] cherchent à contourner les limites de l'interface dans la conception d'un tel système en y introduisant l'écriture manuscrite. Les retours d'utilisation d'un tel système mettent en exergue l'avantage de l'utilisation de l'écriture manuscrite dans le cadre des mathématiques, mais aussi les difficultés rencontrées dans l'interprétation de ces mêmes tracés, introduisant de potentielles erreurs dans les feedbacks réalisés

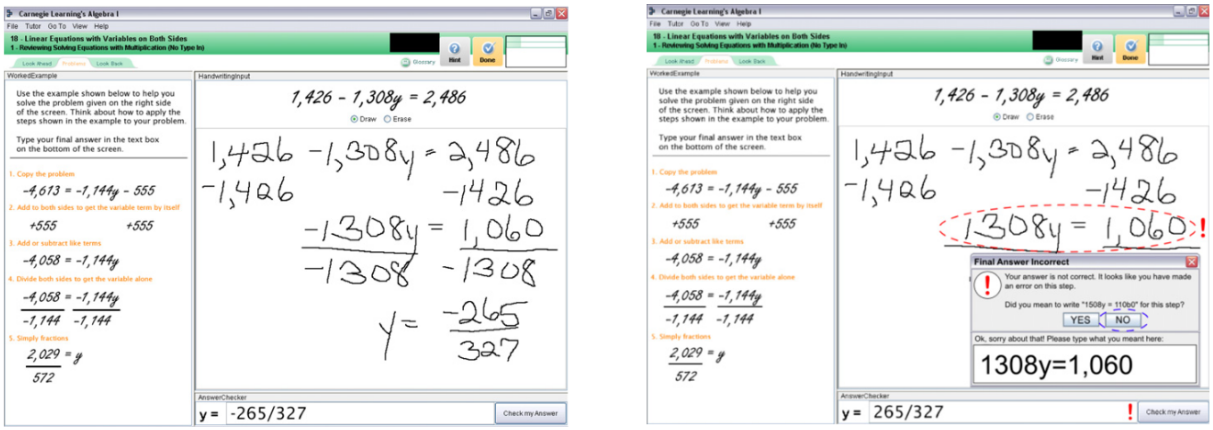

FIGURE 2.4 – Capture de l'interface [AYK12] permettant la saisie manuscrite d'un problème mathématique, et la correction de la saisie en cas d'erreur d'interprétations (droite).

aux élèves. Si l'intérêt d'une telle interface est mis en avant, le module apprenant se doit d'éviter les erreurs d'interprétations afin de réaliser l'implémentation d'un STI satisfaisant dans le processus d'apprentissage. Les auteurs proposent ainsi dans [AYK12] une implémentation d'un tel système, tout en permettant aux élèves de corriger des erreurs de reconnaissance du système par la saisie par clavier de sa réponse (figure 2.4). Si ce processus permet de corriger des erreurs d'interprétation, il entre cependant en conflit avec le processus normal de résolution de l'élève. On peut noter la capacité du système à utiliser la connaissance a priori de l'exercice pour guider cette tâche de reconnaissance, une approche intéressante que nous aborderons dans la suite de ce manuscrit.

Cette complexité d'interprétation de l'écriture manuscrite a limité le développement et la propagation des STI pour les mathématiques. L'application MyBlee (figure 2.1) étant la seule à notre connaissance à proposer l'écriture manuscrite dans le cadre des mathématiques en primaire, mais avec une saisie pré-casée très contraignante. La nécessité d'écrire chaque symbole dans des boîtes prédéfinies correctement alignées limite la transférabilité vers le papier/crayon qui ne disposera pas de tels guidages.

2.1.4 Bilan

Dans cette section, nous avons abordé les fondements pédagogiques qui motivent notre projet. Les outils numériques, à condition qu'ils ne causent pas une surcharge cognitive et permettent un transfert des connaissances entre l'outil et les méthodes traditionnelles sont intéressants et permettent de réaliser des feedbacks complets et personnalisés pour chaque

élève. Cependant disposer d'une interface sans contrainte pouvant reproduire l'espace de liberté d'un élève pour la pose d'opérations met en évidence les besoins de robustesse de l'IA qui est au coeur du STI pour interpréter les saisies manuscrites réalisées sans contraintes de l'élève.

Pour répondre à ces besoins, nous avons conçu et développé un système complet de reconnaissance et d'interprétation de tracés manuscrits afin d'offrir aux élèves une interface répliquant une feuille de papier permettant un transfert de connaissances immédiat entre la réalisation sur tablette numérique et sur papier. Ces travaux permettent de mettre en place des feedbacks de correction ou de guidage afin d'accompagner les élèves dans leur apprentissage et leur progression.

2.2 Reconnaissance à la volée de tracés manuscrits

Nous nous intéressons maintenant à l'axe principal des travaux de thèse permettant d'aboutir à un système de tuteur intelligent pour les mathématiques : la reconnaissance à la volée de tracés manuscrits composant une opération arithmétique. Nous avons précédemment montré l'intérêt de disposer d'une interface libre afin d'éviter une surcharge cognitive et de faciliter le transfert des compétences de résolution d'opérations entre la tablette et le papier. Une opération arithmétique est représentée par une structure en deux dimensions, nous présentons donc les concepts généraux relatifs au domaine de l'interprétation d'un document structuré, puis nous présenterons les travaux concernant la reconnaissance en-ligne d'expressions mathématiques, qui est le domaine de recherche classique le plus proche du notre. Nous y aborderons l'ensemble des problématiques et des méthodes liées, afin d'orienter nos choix technologiques et contributions présentés dans la suite du manuscrit.

2.2.1 Définitions générales

Différences entre reconnaissance et analyse

Pour commencer, il est nécessaire de bien comprendre les différences fondamentales entre l'objectif de reconnaissance de tracés manuscrits et l'objectif d'analyse d'opérations arithmétiques que nous souhaitons traiter.

Dans le cas "historique" de la *reconnaissance* de tracés manuscrits, l'objectif est de reconnaître des tracés dans un contexte de saisie connu. Le prérequis est que les saisies

correspondent à la tâche de reconnaissance définie et que cette saisie est bien formatée. Ce concept est d'autant plus important dans le cadre de documents structurés. Il est supposé que le document en question est correctement formé, ainsi différentes règles peuvent être définies afin de guider la reconnaissance de l'ensemble du document. Cependant aucune information n'est à disposition quant aux tracés en eux-mêmes, on ne sait pas ce que l'auteur a saisi. Dans le cadre d'une expression mathématique, il est usuellement considéré que cette dernière est correctement formatée et respecte les règles mathématiques connues. Cette information permet de guider la reconnaissance, en interdisant au système de l'interpréter comme une expression qui ne serait pas mathématiquement correcte.

Dans notre contexte *d'analyse* d'opérations arithmétiques, nous avons connaissance d'une consigne donnée par l'enseignant : nous disposons donc d'une information supplémentaire sur la nature de la saisie attendue. Cependant, la saisie de l'élève qui est en apprentissage peut être incomplète, contenir des erreurs dans la résolution, ne pas respecter diverses règles algorithmiques ou inhérentes à la pose d'opérations arithmétiques. Ainsi, d'un côté nous pouvons nous appuyer sur les connaissances a priori pour potentiellement guider ou corriger des erreurs de reconnaissance, mais nous ne pouvons pas compter sur la bonne composition de l'opération et nous ne devons pas uniquement prendre en compte ce qui était attendu, au risque de privilégier les attentes à la réalisation de l'élève.

Même si ce n'est pas directement notre contexte, nous commencerons par voir des méthodes de reconnaissance qui se basent sur un contexte de saisie sans informations a priori, mais en présupposant le respect de la structure mathématique. Nous présentons les différentes stratégies existantes qui pourront inspirer pour la reconnaissance d'opérations arithmétiques potentiellement mal structurés. Nous détaillerons par la suite une stratégie nous permettant de tirer bénéfice de connaissances sur la consigne pour guider l'analyse.

Document structuré

Un document structuré est caractérisé par la présence d'une structure établie, définie par un ensemble de règles de construction. Le document peut être composé de symboles tels qu'un ensemble de chiffres ou de lettres mais aussi de formes prédéfinies spécifiques au langage. Les relations spatiales bidimensionnelles entre ces différents éléments sont basées sur des conventions établies, tel que l'alignement (horizontal et vertical) de nombres au sein d'une opération. Ces relations donnent une information importante permettant d'interpréter le document. Le domaine des documents structurés est constitué d'une multitude de formats : diagrammes, circuits électriques, tableaux... Deux typologies de documents

sont distinguées : *"hors-ligne"* et *"en-ligne"*. Dans le cas d'un document "hors-ligne", on dispose de l'image scannée d'un document papier alors que dans le cas "en-ligne", la donnée est représentée sous une forme numérique dynamique, c'est-à-dire de séquences de trajectoires ordonnées correspondant aux mouvements du stylet sur la tablette. Nous allons nous intéresser à ces deux typologies de documents qui peuvent conduire à des approches de reconnaissances différentes.

Création d'un document structuré

Il existe deux manières d'aborder la création d'un document numérique structuré. Il est possible de fournir à l'utilisateur un logiciel proposant une interaction "WIMP" (Windows, Icônes, Menus, Pointer). L'utilisateur peut sélectionner à partir d'un menu et d'un ensemble d'options des symboles à déplacer sur l'interface afin de constituer un schéma. De nombreux tuteurs mathématiques sont basés sur un tel fonctionnement afin de proposer une interface intelligible permettant à l'élève d'y inscrire et résoudre son opération mathématique. Cependant, comme nous l'avons abordé dans la section précédente, une telle interface basée sur une saisie précasée dans une approche d'apprentissage n'est pas complètement adaptée car elle diffère trop de la manipulation papier/crayon rencontrée lors de l'apprentissage de la pose d'opérations arithmétiques.

Nous nous intéressons donc assez naturellement au deuxième type de création de documents structurés, qui passe par la saisie manuscrite de l'utilisateur. Nous distinguons alors deux types de saisies, la première peut se réaliser sur papier, à la suite duquel le document est numérisé et interprété. On représente alors ce document par un signal "hors-ligne", une image, c'est-à-dire une matrice de pixels. L'objectif de l'interprétation est de retrouver dans cette matrice de pixels les tracés du document afin de les segmenter, de les reconnaître, puis de les agencer les uns par rapport aux autres par des contraintes spatiales. La deuxième manière de créer un document structuré à partir de tracés manuscrits correspond à une saisie "en-ligne" du signal. L'élève utilise un stylet (ou à défaut, son doigt) sur l'écran tactile d'une tablette numérique. Le signal est ainsi représenté par une séquence de trajectoires ordonnées temporellement et dont le début (pose du stylet) et fin (levée du stylet) sont enregistrés. C'est sur ce second type de documents que nous allons travailler. Il est néanmoins intéressant de considérer les deux supports dans l'état de l'art. Nous allons donc voir dans la prochaine section les différentes méthodologies qui concernent l'interprétation de ces deux types de documents.

Interprétation de documents structurés

On distingue à l'instar d'un signal hors-ligne deux approches concernant l'interprétation d'un document en-ligne.

— **Approche a posteriori** : l'interprétation ne se fait qu'après la réalisation complète du document à la demande de l'utilisateur. L'avantage d'une telle approche est la disponibilité de l'intégralité de la construction du document, et donc des relations spatiales entre les éléments composant ce dernier. La disponibilité de tout le contexte structurel aide à l'interprétation des tracés. On peut par exemple citer les travaux de [GLA12] sur la reconnaissance de plan 2D manuscrits. Cependant une erreur d'interprétation d'un tracé peut engendrer des erreurs en cascade.
— **Approche à la volée** : les tracés sont interprétés au fil de l'eau, et ainsi une analyse est réalisée à chaque nouveau tracé ou lorsqu'un ensemble de tracés est ajouté sur le document. Il est ainsi possible de fournir à l'utilisateur cette interprétation qu'il pourra valider ou non. Ainsi, si une erreur est commise, l'utilisateur peut directement redéfinir son tracé en amont pour y corriger l'erreur, afin que cette mauvaise interprétation ne pénalise pas le reste du document. Dans un contexte applicatif d'apprentissage, les travaux dans [Kri20] traitent de l'apport et de l'interaction du système au cours de la résolution d'un exercice de géométrie. Il faut cependant noter qu'impliquer l'utilisateur pour corriger une erreur d'interprétation peut présenter une surcharge cognitive qui peut entraver la fluidité de la saisie.

La composition à main levée est la plus pertinente, puisqu'elle permet de donner un degré de liberté totale à l'élève dans la résolution du problème. Dans un processus continu d'apprentissage, il est plus intéressant de mettre à disposition un système en-ligne afin de proposer des retours à l'élève sur la même interface de saisie, plutôt que de couper sa résolution par la prise d'image pour une interprétation différée. On peut ainsi interagir par des feedbacks graphiques immédiats qui vont venir s'afficher en superposition de la saisie de l'élève afin d'apporter une aide et des retours nécessaires. Enfin, concernant l'approche a posteriori ou à la volée, ces deux approches sont pertinentes en fonction du contexte et des besoins d'apprentissage de l'élève. Nous allons présenter un ensemble de travaux concernant l'interprétation "en-ligne" a posteriori d'expressions mathématiques structurées, bien que ces derniers peuvent rester pertinent pour un traitement à la volée.

2.2.2 Reconnaissance en-ligne d'expressions mathématiques manuscrites

Nous venons de nous intéresser à la composition de documents et à l'intérêt des travaux de recherche pour la reconnaissance de documents manuscrits. Nous spécifions dans cette section l'état de l'art dans le domaine de la reconnaissance en-ligne d'expressions mathématiques. Les expressions mathématiques telles que traitées dans l'état de l'art sont très proches structurellement de nos opérations arithmétiques, puisque toutes les deux sont représentées par un ensemble de symboles mathématiques dont les relations spatiales représentent l'agencement de l'opération. L'objectif est donc de décrire les différentes approches observées pour la mise en place d'un système de reconnaissance. Il est nécessaire de se rappeler que les stratégies de reconnaissance présentées par la suite doivent reconnaître une expression mathématique qui est inconnue mais supposée bien écrite et bien structurée. Nous verrons comment cette hypothèse permet de guider la reconnaissance au travers de règles de composition afin de produire une interprétation mathématiquement correcte des saisies.

Tâches et stratégies

La reconnaissance d'expressions mathématiques a suscité de l'intérêt dans les années 1960, mais ces premiers travaux se sont concentrés sur la reconnaissance hors-ligne de documents imprimés. L'évolution des écrans tactiles et de la manipulation de stylets dans les années 2000 a permis au domaine de la reconnaissance en-ligne de croître considérablement. L'objectif de la reconnaissance d'expressions mathématiques est de transformer l'ensemble de tracés manuscrits en une représentation sous forme d'arbre (MathML, LaTeX), et se décompose en deux tâches principales : la reconnaissance de symboles et l'analyse structurelle. La reconnaissance de symboles va comporter les sous-tâches de prétraitement, de segmentation de tracés et de classification de symboles. La deuxième étape d'analyse structurelle consiste à agencer les symboles entre eux suivant des relations spatiales mathématiques prédéfinies, et à potentiellement élaguer ces relations pour retrouver la structure principale de l'opération.

Dans [ZZR21], trois principales stratégies sont identifiées :

— Les stratégies séquentielles (généralement à base de graphes ou d'arbres) sont caractérisées par la séparation de chaque étape de reconnaissance, et l'utilisation des résultats d'une étape précédente pour l'étape suivante.

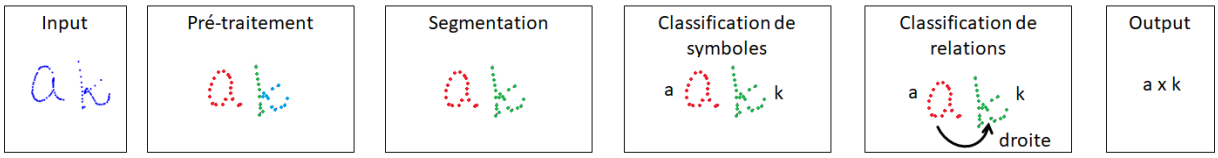

FIGURE 2.5 – Sous-tâches séquentielles à partir d'un ensemble de tracés permettant d'aboutir à la reconnaissance d'une expression mathématique.

— Les stratégies intégrées (généralement à base de grammaires) vont générer un ensemble de symboles hypothèses, et c'est l'analyse structurelle qui se chargera de sélectionner les meilleurs symboles candidats à l'aide de grammaires définies.
— Les stratégies de bout-en-bout (basées sur des réseaux de neurones) vont directement transformer les tracés en une notation mathématique adéquate. L'apprentissage des différentes tâches est conjoint et global, sans prétraitement initial de la saisie.

Afin d'aboutir à la reconnaissance d'une expression mathématique, plusieurs tâches sont traitées par les stratégies à base de graphes ou de grammaires. La figure 2.5 représente l'agencement séquentiel de ces sous-tâches.

Pré-traitement des tracés. La première étape pour la reconnaissance des tracés manuscrits consiste à normaliser l'ensemble des données traitées. En effet les données saisies pouvant provenir de multiples sources, cela permet d'améliorer significativement les performances de reconnaissance en diminuant les variations entre saisies. Un grand ensemble de méthodes vont procéder à la suppression de points dupliqués, au lissage des tracés, à un nouvel échantillonnage pour diminuer l'impact de la fréquence de saisie d'une tablette, et à la normalisation de la taille des tracés au sein d'une opération. Certaines méthodes appliquant des modèles de langue sur les tracés, un réordonnancement de ces tracés est aussi proposé afin que l'ordre de choix des tracés de l'utilisateur n'impacte pas la reconnaissance.

Segmentation des tracés. L'objectif de segmentation de tracés est de regrouper des ensembles de tracés afin de les traiter en tant que symbole. Cette tâche est souvent résolue par des Modèles de Markovs Cachés [HBT96] ou par l'utilisation d'un Arbre de Couverture Minimal [ZBC02]. Une segmentation utilisant des caractéristiques géométriques entre les paires, les tracés voisins et les formes, en utilisant un algorithme AdaBoost [HZ13] ou l'utilisation d'un classifieur type Support Vector Machine (SVM) s'appuyant sur un ensemble de caractéristiques géométriques sont aussi proposés [OS04]. D'autres caracté-

ristiques comme les "Shape Context features" [HZ16a] sont proposées pour résoudre cette problématique de segmentation.

Classification des symboles. Lorsque les tracés sont regroupés en hypothèses de symboles, l'objectif est d'en déterminer la classe d'appartenance. Historiquement la reconnaissance d'écriture se basait sur des méthodes standards utilisant les Modèles de Markovs Cachés, mais la majorité des méthodes se basent aujourd'hui sur des réseaux tel que des réseaux de neurones à convolution (CNN) [RPA16], ou des "Bi-directionnal Long Short-Term Memory" (LSTM) [ZMV17] qui ont la particularité d'analyser des séquences de symboles, et ainsi d'utiliser le contexte de chaque symbole afin d'en déterminer la classe d'appartenance.

Analyse structurelle. Enfin la dernière tâche pour la construction d'une expression mathématique est l'analyse structurelle, pour déterminer les relations spatiales entre symboles. Pour la classification de relations spatiales, les classes recherchées pour les expressions mathématiques sont : Droite, Exposant, Indice, Au-dessus, Dessous, À l'intérieur. Les méthodes se basent généralement sur des positionnements relatifs de boîtes englobantes, en fonction de la nature et de la taille des symboles [ZBC02]. Des caractéristiques géométriques et de formes sont généralement utilisées [SKC15]. Grâce à cet agencement, il est possible de fournir une interprétation de l'expression.

Stratégies à base de graphes ou d'arbres

Les **stratégies à base de graphes** se basent sur des caractéristiques géométriques entre symboles, et des règles permettent de détecter et corriger des erreurs d'interprétation. Les opérations seront représentées par des graphes (ou arbres) dont les noeuds (ou feuilles) sont les symboles mathématiques, et les liens sont les relations mathématiques entre symboles. L'objectif est de réduire ce graphe à un graphe de couverture minimale (figure 2.6). Cette représentation élimine l'ensemble des relations mathématiques redondantes entre symboles pour former une expression mathématique en partant d'un symbole racine afin que chaque symbole ne soit enfant au plus que d'une seule relation mathématique. Cette réduction est généralement réalisée en se basant sur les scores attribués à chaque relation du graphe pour élaguer ce dernier et conserver l'interprétation la plus vraisemblable.

Parmi les premiers travaux s'attaquant à la reconnaissance d'opérations mathématiques, on peut citer [Mat99] qui a créé un premier graphe de tracés en reliant ces derniers en fonction des distances euclidiennes entre boîtes englobantes pour proposer une pre-

FIGURE 2.6 – Arbre de couverture minimale entre tracés générés à partir des distances euclidiennes entre boîtes englobantes. [Mat99]

mière segmentation entre tracés proches. Les boîtes englobantes sont définies à partir des positions des tracés pour déterminer les relations entre différents symboles. Un symbole clé, généralement le plus grand de l'opération, est sélectionné et considéré comme "racine" de l'opération. Chaque symbole est ensuite regroupé en fonction de sa relation par rapport au symbole clé, puis les autres symboles sont traités de manière récursive jusqu'à traiter l'entièreté de l'opération. Une grammaire est par ailleurs proposée pour proposer une sélection de symboles adéquats qui n'enfreint pas de règles mathématiques. Chaque classe de symboles est par ailleurs assignée à un type (ascendant, descendant ou simple) pour déterminer le corps du symbole afin de mieux discriminer les relations mathématiques.

Par la suite dans [ES01] est présenté un réseau de liens virtuels (figure 2.7). L'objectif est d'affiner la définition des relations entre symboles en calculant les positions relatives des symboles les uns par rapport aux autres en fonction de leur nature (lettres, chiffres, intégrales...). Différentes zones sont définies en fonction des paires de symboles observées (lettre-lettre, lettre-chiffre...) pour déterminer l'appartenance d'une paire de symboles à une relation en fonction des valeurs obtenues. Un ensemble de règles permet de construire l'arbre de couverture minimale représentant l'opération mathématique.

[ZBC02] proposent d'étendre le concept de relations mathématiques en définissant pour chaque typologie de symboles des zones adéquates pour chaque relation mathématique. En observant l'agencement des symboles, si la base d'un symbole est comprise dans une de ces zones alors une relation lui est attribuée (figure 2.8). La construction de l'opération se réalise sous la forme d'un arbre. Une première ligne de base contenant un agencement de plusieurs symboles alignés horizontalement est définie, puis pour chacun de ces symboles une nouvelle recherche est effectuée pour les autres types de relations mathématiques afin de constituer l'arbre final.

D'autres méthodes présentent différentes approches pour définir le symbole de départ constituant l'opération soit par un ordonnancement temporel ou spatial [VHH08]. Les

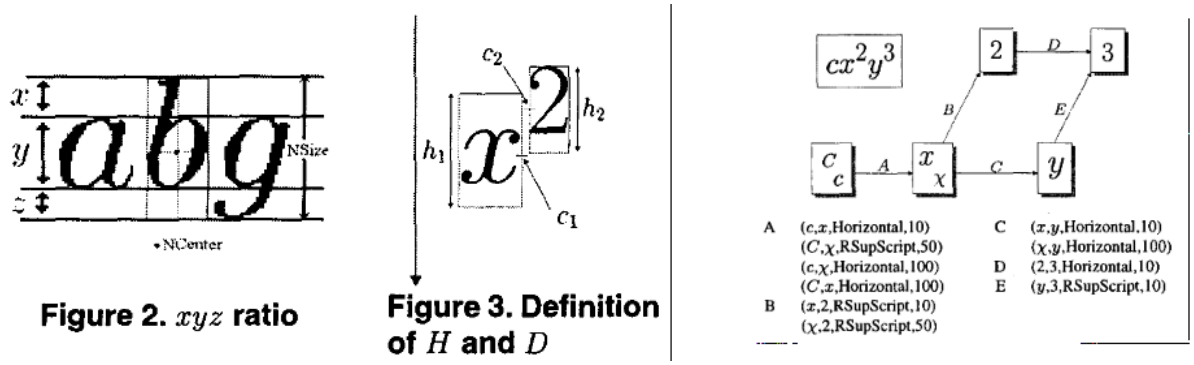

FIGURE 2.7 – Un réseau de lien virtuels, avec des relations définies entre symboles et leur coût respectif. [ES01]

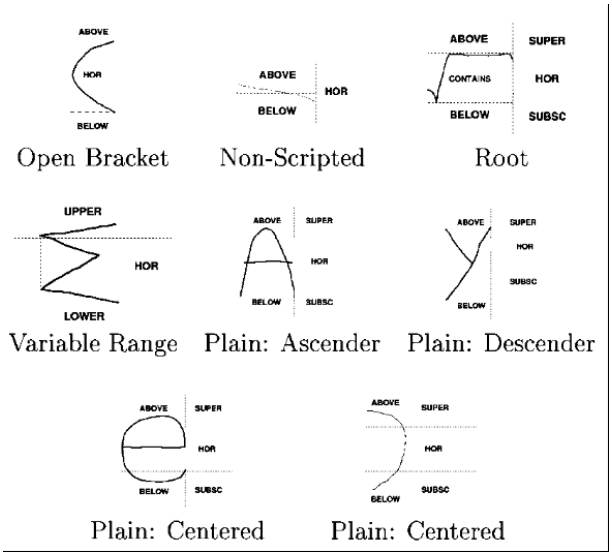

FIGURE 2.8 – Régions définies pour certains types de symboles afin de déterminer l'appartenance d'autres symboles à de potentielles relations mathématiques. [ZBC02]

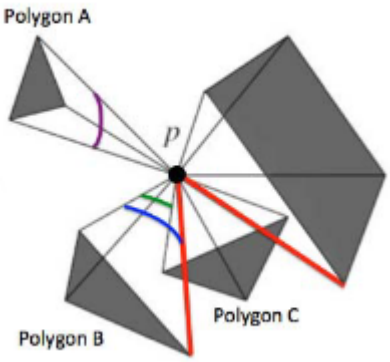

FIGURE 2.9 – Angle de vision à partir du centre d'un tracé P vers l'ensemble des formes observées dans son entourage pour déterminer les visions non obstruées. [HZ16a]

relations mathématiques permettant la constitution de graphes ou d'arbres représentant l'opération sont aussi définies plus précisément en reprenant les formes et tailles des symboles ou en y incluant des zones permettant de définir les relations moins strictes. Plus récemment [HZ16a ; HZ16b] proposent une approche basée sur la notion d'angle de vision pour constituer un premier graphe de recherche sur lequel sera construite l'opération mathématique. Ils introduisent le concept de ligne de mire afin de relier des tracés même très éloignés pour ne pas éliminer prématurément de potentielles relations qui seraient ignorées à cause de distances variables ou de superposition de symboles. Des angles de visions entre chaque tracé sont formés et permettent la création de liens entre tracés si un angle de vision au moins n'est pas obstrué par des tracés plus proches (figure 2.9).

Pour déterminer l'appartenance de tracés à un même symbole, des "contextes de forme Parzen" sont calculés entre les paires de symboles et le reste de l'opération pour observer les zones et fréquences des points des tracés dans l'entourage. L'objectif de cet ensemble de caractéristiques est de définir dans l'espace la fréquence d'apparition et la position relative des points. En partant du centre de la paire de tracés, un espace est défini et découpé en zones, définies par des angles et des distances. Chaque point d'un tracé contribue dans la zone dans laquelle ce point apparaît. La figure 2.10 illustre ce concept. Le calcul est effectué 3 fois, deux contextes de formes sont extraits pour chaque tracé candidat d'un symbole (figure 2.10 (1) et (2)), et un dernier est effectué relativement aux autres tracés de l'opération (figure 2.10 (3)). Deux stratégies existent pour attribuer une valeur à chaque zone, la première consiste simplement à compter les points apparaissant dans chaque zone, tandis qu'une autre approche applique une valeur dans toutes les zones en fonction de

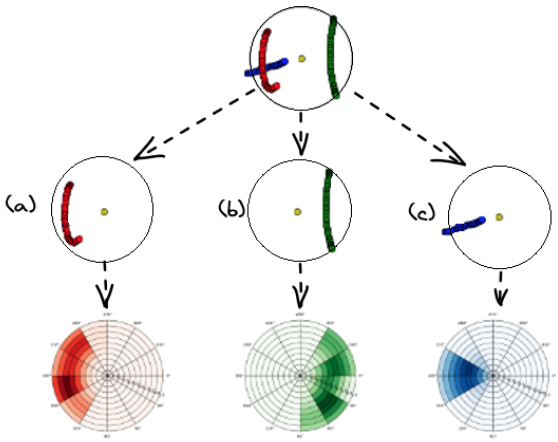

FIGURE 2.10 – Extraction du contexte de formes pour deux tracés. Un ensemble de zones angulaires sont définies pour déterminer l'appartenance des points des deux tracés dans chaque zone, et de noter la fréquence d'apparitions des points des autres tracés de l'opération. On considère ici le tracé vertical du "plus" (a) et le tracé vertical du 1 (b). Le tracé horizontal (c) quand à lui fait partie des tracés en dehors de la paire considérée. Les différentes zones montrent la fréquence d'apparition des points. [HZ16a]

la distance entre le point et la zone considérée. Cela permet de définir plus précisément l'emplacement des tracés au sein de l'espace, mais contribue à augmenter de manière non négligeable la quantité de calculs nécessaires pour l'extraction de ces caractéristiques. Ces mêmes caractéristiques sont utilisées pour déterminer les relations mathématiques entre symboles, et un algorithme est appliqué sur ces relations attribuées pour produire le graphe de couverture minimale.

Ces méthodes permettent d'obtenir une représentation temporaire complète représentant l'opération et l'agencement de nombreux symboles entre eux. La réduction en un graphe ou arbre de couverture minimal permet de créer une opération sémantiquement correcte. Elles n'appliquent pas ou peu de règles pour la création de la représentation initiale, ce qui leur permet une malléabilité importante même lors d'opérations mal formées. Cependant ces méthodes ne peuvent pas utiliser le contexte global de l'opération, et on observe donc des scores de reconnaissance relativement faibles par rapport à d'autres approches qui considèrent l'ensemble de l'opération avec des règles ou des informations contextuelles, que nous allons voir par la suite.

Stratégies à base de grammaires

Les **stratégies à base de grammaires** quant à elles utilisent un ensemble de règles de production pour l'analyse. L'analyse repose sur les résultats de reconnaissance de symboles, sur les règles de grammaire, sur les scores de positionnement et sur des modèles de langues. Cela permet de privilégier une interprétation de l'expression qui est sémantiquement correcte. Ainsi des confusions de relations entre symboles mathématiquement impossibles seront rejetées. Cependant, si l'opération de base saisie n'est pas correcte mathématiquement, alors les scores de reconnaissance seront pénalisés puisque l'on privilégiera une interprétation correcte. Ces stratégies sont basées sur des grammaires dites context-free. Les plus utilisées sont les 2D Stochastic Context-Free Grammar (SCFG), appliquant une version probabiliste de l'algorithme Cocke-Younger-Kasami (CYK) [You67]. Certaines méthodes proposeront de réaliser simultanément la reconnaissance de symboles et l'analyse structurelle avec ces contraintes en appliquant conjointement la grammaire et un modèle de reconnaissance de symboles. L'analyse d'une telle grammaire étant complexe, plusieurs travaux proposent d'en réduire la complexité ou de supprimer des hypothèses impossibles dans l'arbre de recherche.

Pour compenser les accumulations d'erreurs qui peuvent se produire en enchaînant un ensemble de modèles sans information contextuelle, en 2006, [Yam+06] proposent d'utiliser une grammaire stochastique sans contexte appliqué sur l'ensemble des tracés pour reconnaître l'opération mathématique. L'idée au travers de règles de grammaire est notamment de pouvoir éviter les erreurs d'interprétations de symboles proches (comme 'C' et '(') en guidant l'espace de recherche. Si au sein de l'opération un symbole ')' est reconnu, alors l'hypothèse que le symbole observé plus tôt soit un '(' au lieu d'un 'C' va être privilégié afin de constituer une opération valide. Les auteurs introduisent une grammaire incluant notamment des contraintes comme la présence d'un jeton "parenthèse gauche" contraint la présence d'un jeton "parenthèse droite". L'algorithme CYK est utilisé afin de réaliser l'analyse syntaxique de la grammaire. Un premier score est proposé pour chaque tracé, puis dans les étapes suivantes de l'algorithme les paires de symboles sont groupés en un ensemble d'expressions valides pour la grammaire. Les probabilités de chaque expression sont déterminées à la fois par les scores de classification de symboles ainsi que par les scores de positionnement déterminés par des zones définies sur les boîtes englobantes des symboles. Lorsque tous les tracés sont traités, l'expression avec la plus forte probabilité est conservée.

Cette approche permet donc de résoudre en une passe la reconnaissance de l'opération

tout en y appliquant des contraintes sémantiques. Cependant cet algorithme est contraint par la temporalité de chaque tracé, puisque les tracés considérés sont adjacents temporellement. De plus, l'algorithme est complexe à réaliser car à chaque étape un grand nombre d'expressions temporaires, et donc de combinaisons d'expressions, peuvent être considérées, rendant l'application de cet algorithme sur de grandes opérations complexes. Dans [LVN14], l'équipe propose de traiter spécifiquement les sous-expressions qui sont observées comme ayant un ordre de saisie qui peut varier en introduisant de nouvelles règles dans la grammaire pour prendre en compte ces dernières. Ils proposent aussi des concepts plus élaborés sur les typologies des eux (ascendant, descendant...) pour déterminer les relations entre symboles et donc les probabilités qui en découlent. D'autres méthodes utilisent des approches similaires en utilisant l'algorithme CYK [ASB14 ; AMV14], mais elles souffrent toutes des mêmes problèmes de complexité sur de grosses opérations et ne peuvent reconnaître des opérations mal formées.

Pour compenser la complexité de l'analyse, [SKC15] proposent d'utiliser le résultat d'une reconnaissance optique de caractères (OCR) pour appliquer la recherche uniquement sur l'ensemble déjà segmenté des symboles. Cependant des erreurs de l'OCR induiront par conséquent des erreurs dans la suite de l'analyse. Ils proposent aussi par ailleurs de ne conserver à chaque étape qu'un nombre limité d'expressions valides (celles avec la probabilité la plus forte) pour éliminer des hypothèses invraisemblables.

Les méthodes à base de grammaires introduisent par l'application de règles un ensemble de contraintes que doit respecter l'opération, permettant de corriger des erreurs de reconnaissance qui constitueraient une expression non valide. Cependant, le processus d'analyse de la grammaire, la construction de la grammaire et l'obligation pour l'opération finale de respecter cette grammaire définie contraignent le domaine d'application de telles méthodes. Dans notre contexte de travail de thèse, l'objectif est de pouvoir analyser des opérations potentiellement mal formées structurellement ou sémantiquement. Il faudrait ainsi concevoir une grammaire dont les règles puissent accepter des structures d'opérations incorrectes avec entre autres des suites de symboles inadéquats. Or, l'intérêt même de ces règles est de pouvoir restreindre l'espace de recherche afin de guider ou corriger de potentielles erreurs de reconnaissance grâce à des règles strictes, limitant ainsi l'intérêt d'une telle grammaire. C'est pourquoi nous nous orienterons plutôt vers des stratégies à base de graphes que nous présenterons dans le chapitre 3.

Stratégies de bout-en-bout

Les stratégies bout-en-bout traitent l'évaluation structurelle en parallèle des autres tâches. La structure de l'opération va être construite en transformant l'image produite par l'ensemble des tracés en une opération LaTeX qui représente déjà la structure finale de l'opération.

Ces méthodes sont adaptées à une reconnaissance hors-ligne d'expressions mathématiques, mais peuvent parfaitement être appliquées aux opérations en-ligne. En effet, les méthodes que nous présenterons par la suite se basent toutes sur la transformation du signal en-ligne en un signal hors-ligne pour y appliquer le réseau appris afin de générer la reconnaissance de l'expression mathématique.

Une telle approche a d'abord été proposée en 2017 [Den+17]. Un réseau de neurones à convolution commence par extraire des caractéristiques du signal hors-ligne. Puis deux réseaux de neurones récurrents sont utilisés pour encoder puis décoder l'expression en une opération (au format LaTeX) valide. Devant le succès de la méthode, plusieurs améliorations de l'architecture du système sont proposées [Zha+17 ; ZDD18a ; ZDD18b], en y instaurant notamment un système d'attention afin d'utiliser le contexte de la reconnaissance précédemment atteinte pour permettre une meilleure interprétation de l'environnement. Ces méthodes obtiennent les meilleurs résultats de reconnaissances sur des expressions mathématiques dans les dernières compétitions scientifiques [Mou+16 ; Mah+19], de l'ordre de 90% de reconnaissance sur l'expression complète.

Cependant, la tâche se limite à la transformation d'un ensemble de tracés en une expression valide. Ainsi, chaque symbole est au plus enfant d'un seul symbole mathématique, on perd donc une part importante de l'information structurelle, présente dans les représentations intermédiaires sur les graphes vues précédemment. De plus, de telles méthodes ont besoin des bases d'apprentissage conséquentes, nécessitant ainsi la constitution et l'annotation de volumes de données importants. De multiples approches sont proposées pour compenser le manque de données d'apprentissage au travers de la génération de données [LIN19]. Mais dans notre contexte d'utilisation, cela est d'autant plus marquant qu'il serait nécessaire d'observer et d'annoter de multiples typologies d'erreurs afin de ne pas pénaliser les scores de reconnaissance.

2.2.3 Bilan

Le domaine de la reconnaissance en-ligne d'expressions mathématiques a considérablement progressé ces dernières années avec l'exploitation des réseaux de neurones profonds à base de l'architecture Encodeur-Decodeur. Les performances brutes de reconnaissances d'opérations atteignent des seuils très satisfaisants sur les benchmarks internationaux. Cependant plusieurs limitations découlent de ce type de réseau. Ces méthodes se concentrent sur la tâche de reconnaissance, et à cette fin transforment l'ensemble des tracés en une représentation simple et limitée de l'opération. On perd donc une information très importante sur l'agencement des symboles entre eux. Si cela n'est pas nécessaire pour une tâche de reconnaissance, dans notre contexte, il est nécessaire d'interpréter finement l'ensemble des relations, pour qualifier la nature des erreurs de l'élève et produire des feedbacks pertinents. En effet, pour des opérations arithmétiques, connaître l'agencement de tous les nombres entre eux est important pour en détecter les erreurs de pose. Ces modèles complets ne peuvent être appliqués dans un contexte en-ligne et incrémental, où il serait nécessaire de réappliquer le modèle à l'ensemble de l'opération à chaque incrément. Dans le cadre d'une application en temps réel, cela peut entraîner des complexités dans les temps de calcul nécessaire à l'application du modèle. Des limites seraient aussi observées dans le cadre d'opérations mal formatées, et il y a un besoin d'une grandes bases d'apprentissage annotées afin d'obtenir des modèles performants. Enfin, notre tâche d'analyse ne se contente pas de résoudre la tâche de reconnaissance : il est nécessaire de pouvoir extraire une structure complète représentative de l'opération afin de pouvoir confonter la solution d'un élève à une solution attendue.

Au contraire, les stratégies basées sur une segmentation explicite à base de graphes sont plus adaptées à notre problématique. Elles présentent des résultats inférieurs en termes de score de reconnaissance, cependant elles permettent de représenter l'ensemble des symboles et la structure complète de l'opération. De plus, contrairement aux approches "end-to-end", il est possible de construire progressivement la représentation au fil de la saisie afin d'éviter une trop grande attente lorsque l'élève a fini d'écrire son opération. Nous présenterons cette stratégie dans la suite du manuscrit. La possibilité de produire une telle représentation permet de développer une analyse fine des erreurs commises par l'élève qui est nécessaire pour produire des feedbacks personnalisés qui le guideront dans son apprentissage. De plus, il nous est possible dans notre contexte d'analyse d'insérer des connaissances attendues afin de guider la reconnaissance et donc corriger les problématiques de reconnaissances imprécises sur des écritures dégradées d'enfant. Nous avons ainsi

sélectionné la stratégie basée sur des graphes de par la richesse d'une telle représentation.

La saisie de l'élève sera traitée pour produire une première interprétation sous forme d'un graphe représentant l'opération arithmétique, avec les symboles mathématiques la contenant et l'ensemble de leurs relations. Par ailleurs, en utilisant la consigne fournie par l'enseignant, nous générerons un graphe idéal, construit avec les règles algorithmiques correspondant à l'opération attendue. Ces deux représentations similaires pourront être comparées à la fois à des fins d'analyses pour produire les feedbacks nécessaires, mais aussi pour potentiellement remettre en question notre première interprétation afin de générer de nouvelles interprétations si des différences particulières sont observées entre les deux représentations. Nous nous attardons donc dans le prochain chapitre au principe de mise en correspondance de graphes, afin de pouvoir produire les feedbacks associés à la reconnaissance précédemment vu des tracés en-ligne. Cette mise en correspondance permettra d'obtenir une interprétation de l'ensemble des tracés de l'élève vis-à-vis de l'opération attendue (consigne).

2.3 Mise en correspondance de tracés manuscrits

Notre approche s'appuie sur une représentation structurelle à base de graphes des opérations arithmétiques manuscrites. À partir de cette représentation, nous allons développer une analyse par mise en correspondance entre le graphe des saisies de l'élève et le graphe du modèle expert attendu. Dans le contexte de l'analyse d'opérations arithmétiques d'enfant, cette représentation peut contenir des erreurs telles que de mauvais symboles, des oublis ou des ajouts, une mauvaise pose de l'opération... Il est nécessaire, afin d'obtenir une interprétation fine de l'opération de l'élève, de s'appuyer sur le modèle expert, qui correspond aux attentes de la résolution du problème (consigne). Dans cette section, nous commençons par définir ce principe de mise en correspondance et des problématiques mises en jeu avant de présenter des approches permettant d'y parvenir.

2.3.1 Définitions générales

Définition 3 (Mise en correspondance)**.** Capacité à apparier deux graphes structurels en mettant en relation les noeuds du premier graphe (saisies de l'élève) avec le second (modèle de l'opération attendue). Pour cela, nous allons réaliser une combinaison paire à paire de ces éléments. Il est ainsi possible de comparer la nature, mais aussi les relations observées

entre chacun des symboles afin d'établir un ensemble de différences. Les symboles dans la solution attendue étant définis et représentant des parties spécifiques de l'opération, il sera ainsi possible d'émettre des hypothèses d'interprétation de la réalisation de l'élève.

Cette mise en correspondance repose sur ce que l'on appellera la **GED** (Graph Edit Distance). La GED entre deux graphes G et H représente le coût minimal d'un chemin d'édition P permettant de transformer le graphe G vers le graphe H. Ce chemin d'édition représente un ensemble d'insertion, de suppression et de changement entre noeuds des graphes. Le coût d'un chemin d'édition est représenté par la somme du coût de chacune de ces opérations réalisées pour la transformation du graphe G vers le graphe H. En conservant l'ensemble des changements entre noeuds, on peut ainsi obtenir la mise en correspondance pour une opération arithmétique entre les symboles réalisés par un élève et la solution attendue.

2.3.2 Mise en correspondance de graphes

Il est important de noter que le calcul de GED est un problème NP complet. Il est donc impossible de pouvoir calculer l'appariement entre deux graphes en temps raisonnable dès lors que ces derniers dépassent une dizaine de noeuds. Certaines approches proposent d'accélérer le calcul de la GED, notamment par des approches d'élagage d'arbres de recherche. D'autres méthodes s'appliquent à l'approximation du calcul de la GED permettant de l'effectuer sur des arbres à très grandes dimensions, en pénalisant l'exactitude du résultat et de l'appariement en découlant. Ces méthodes inexactes pertinentes dans le cadre d'un exercice de classification, où l'on peut rechercher et déterminer l'appartenance à une classe d'un graphe par sa proximité relative à un autre ensemble de graphes. Bien que ces stratégies ne puissent être retranscrites directement dans notre contexte où nous recherchons l'exactitude de la mise en correspondance entre la réalisation de l'élève et le modèle attendu, ces approches proposent des recherches sur des sous-ensembles de graphes qui restent pertinentes et seront explorées par la suite. Nous détaillons donc brièvement à la fois les approches de calculs de GED exactes, puis les différentes approches d'approximation, afin d'étudier celles qui peuvent être appliquées à nos graphes d'opérations arithmétiques, pour développer une mise en correspondance optimale.

Approches de calculs exacts de GED

L'approche standard pour le calcul exact de la GED est l'algorithme A*-GED [RFB07] qui se base sur une recherche "best-first" afin de trouver le chemin d'édition optimal. Le processus de recherche au sein de l'arbre est direct. Pour un noeud donné, on calcule la correspondance avec l'ensemble des noeuds de l'autre graphe. En conservant l'ensemble de ces coûts en mémoire, on recherche ensuite la correspondance noeud à noeud à partir de la branche de moindre coût, en remontant dans l'arbre de recherche dès lors que l'ensemble d'appariement de l'arbre dépasse le coût le plus bas observé dans une branche précédente. Ce parcours d'arbre est très lent et coûteux en mémoire dès lors qu'une dizaine ou plus de noeuds sont considérés, car on va parcourir une grande partie de l'arbre de recherche avant d'aboutir au chemin. De plus, l'ensemble des coûts de correspondance au sein des différentes branches parcourues doivent être conservés afin de pouvoir continuer la recherche.

Une amélioration, nommée DF-GED est proposée dans [Abu+15]. Cette amélioration va effectuer une recherche en profondeur sur les noeuds afin de trouver le coût le plus faible. Légèrement plus rapide que l'algorithme A*-GED, il est surtout beaucoup plus efficace en utilisation mémoire car on conserve une très petite partie de l'arbre de recherche à un instant donné. Par la suite, l'algorithme CSI-GED [GH16] a été proposé pour réaliser de la même manière que DF-GED une recherche en profondeur, sur les liens entre noeuds cette fois. Cette méthode était au départ limitée à une application à des coûts d'édition uniforme, et par la suite généralisée pour fonctionner avec des coûts non uniformes [BG20]. Une autre approche utilise un solveur propriétaire BIP-GED [Ler+17]. L'ensemble de ces approches ont été évaluées dans [BG20], et bien que les diverses approches permettent des améliorations sur de petits graphes (< 10 noeuds), dès lors que les graphes atteignent un ordre de grandeur de 15 noeuds ou plus, la majorité des méthodes dépassent les 1000 secondes sur des ordinateurs standards. Les méthodes à base de solveur n'atteignent cette limite qu'à partir de 20 noeuds, mais restent avec un temps d'exécution important néanmoins.

Ainsi, si diverses méthodes cherchant le calcul exact de la GED d'une manière plus optimisée ont été proposées, elles ne peuvent résoudre des exécutions sur des graphes contenant plus de 10 noeuds dans un temps raisonnable pour notre contexte applicatif.

Approches de calculs approximées de la GED

Ceci nous amène donc à la deuxième famille d'approches, principalement utilisée dans le cadre d'analyse de patterns où un appariement exact n'est pas recherché. Ces méthodes se basent sur des heuristiques ou des transformations de graphes permettant d'accélérer grandement les calculs. On présente ci-dessous un bref résumé des principales approches qui peuvent nous intéresser étant donnée la nature de nos graphes d'opérations arithmétiques. Un ensemble plus large de méthodes peut être retrouvé dans [Gao+10] et [Blu+20].

Parmi les approches sur le calcul approximé, il est particulièrement intéressant de mentionner l'approche de "**sous-graphe commun maximum**". L'objectif de ces méthodes est de retrouver un isomorphisme entre deux sous-graphes des graphes principaux, c'est-à-dire trouver, à partir de deux graphes comparés, deux sous-graphes avec une correspondance exacte. Dans notre contexte applicatif, il est possible d'identifier des sous-graphes ayant une signification sémantique dans la réalisation de l'opération : les opérandes présentes dans la consigne, le résultat, ou les opérateurs sont autant de sous-graphes qu'il est possible d'isoler et de traiter. Il deviendrait donc possible d'utiliser des méthodes complètes de calcul de GED sur des ensembles plus petits. Cependant il reste un problème sur des opérations dont ces sous-décompositions resteraient d'une taille trop importante pour ces méthodes.

Ainsi pour pallier aux limites de ces approches, de nombreuses heuristiques ont été proposées pour tenter d'approcher le calcul de la GED. Ces dernières sont présentées, avec leurs limitations dans [Blu+20]. Dans le cadre d'une classification ou d'une recherche de pattern, ces méthodes permettent un calcul rapide de la GED. Certaines de ces heuristiques permettant de calculer une borne inférieure du vrai coût de remplacement, il est possible de les utiliser en parallèle des algorithmes de parcours d'arbres afin d'accélérer la recherche de ces derniers. Notamment, nous pouvons en profiter pour accélérer la recherche d'un sous-graphe commun afin de réduire l'espace de recherche complet.

En utilisant à la fois un arbre de recherche complet, couplé à des heuristiques se rapprochant au maximum de la GED, il nous sera donc possible de dépasser les limitations d'une recherche complète de la correspondance entre deux graphes, permettant la mise en place de cet appariement dans un contexte applicatif immédiat pour l'analyse des opérations arithmétiques.

2.3.3 Bilan

Nous avons présenté, à partir d'une reconnaissance et modélisation des tracés d'un élève sous la forme d'un graphe, différentes méthodes de l'état de l'art permettant de trouver une mise en correspondance entre le graphe des saisies de l'élève, et le graphe attendu du modèle expert qui représente la solution idéale. Chaque graphe contient les informations sur la nature des symboles et leur agencement spatial, et lorsqu'une paire de graphe est mise en correspondance, il est possible de mettre en exergue les différences entre ces derniers. Ces différences peuvent ensuite être interprétées afin d'estimer les erreurs potentielles commises par l'élève et produire des feedbacks adéquats. Si ces méthodes sont efficaces par leur capacité à mettre en relation deux représentations différentes, elles s'en retrouvent cependant limitées par leur complexité d'appariemment. La suite du manuscrit présentera des contributions utilisant le contexte applicatif des opérations arithmétiques posées pour traiter des graphes contenant de nombreux noeuds.

Dans le prochain chapitre, au regard de notre problématique et de l'état de l'art associé, nous présenterons le système conçu et proposé au cours de cette thèse dans son ensemble, afin d'aborder le processus complet et de mettre en exergue les contributions qui seront présentées. Nous y aborderons les différents modules et l'interaction de ces derniers dans une boucle d'analyse entière de tracés manuscrits. Les sections 3.2 et 3.3 présenteront les contributions des deux axes principaux qui sont respectivement la reconnaissance et la mise en correspondance.

Chapitre 3

Analyse d'opérations arithmétiques

Dans ce chapitre, nous reprenons les problématiques mises en avant par le sujet de recherche et les contributions apportées par notre système pour les résoudre. Dans un premier temps, l'ensemble de l'architecture est présenté, afin d'introduire et de comprendre le processus complet d'analyse d'une opération arithmétique. Puis les modules et contributions y sont détaillés afin de présenter le système d'analyse d'opérations arithmétiques proposé.

Sommaire

3.1 Architecture du système d'analyse d'opérations arithmétiques

3.1.1 Rappel des problématiques

Nous avons précisé les différences fondamentales entre l'objectif d'un système classique qui s'attelle à la reconnaissance d'une équation mathématique et notre objectif qui est de procéder à l'analyse d'une opération arithmétique. Pour la reconnaissance, il s'agit de retranscrire une équation mathématique à partir des tracés. Pour l'analyse, il s'agit d'interpréter les tracés à partir d'une consigne afin de créer un ensemble de feedbacks sur l'opération tracée, pour mettre en avant les erreurs et les réussites dans le cadre d'un exercice fourni par l'enseignant. On a donc 3 tâches distinctes à considérer : reconnaissance, interprétation et création de feedbacks. Il est nécessaire d'appliquer une reconnaissance des tracés pour obtenir une représentation adéquate de l'opération. Cette représentation doit être suffisamment riche pour permettre une comparaison entre la production de l'élève et la solution attendue par le professeur. L'interprétation doit pouvoir mettre en relation cette représentation avec la consigne afin de mettre en avant des différences présentes au sein de l'opération par rapport à un modèle attendu. Il sera ainsi possible de gérer la dernière tâche de création de feedbacks compréhensibles pour l'enfant. Au-delà de l'objectif même de l'analyse qualitative, il convient de garder en mémoire qu'il est important de pouvoir compléter cette dernière très rapidement pour garantir des feedbacks immédiats à l'élève. En ce sens, de multiples contributions du système concerneront la capacité à accélérer le processus d'analyse complet tout en ne sacrifiant pas la qualité de ce dernier.

Pour la tâche de reconnaissance, il s'agit de ne pas faire d'erreurs de reconnaissance sur les tracés de l'élève, afin de ne pas propager ces erreurs par la suite sur les feedbacks qui lui seront faits. En effet si un symbole est mal reconnu, alors l'interprétation qui sera réalisée sera erronée. Pour la tâche d'interprétation, il s'agit de pouvoir efficacement produire la comparaison entre la représentation des tracés et la représentation de la consigne. L'utilisation des caractéristiques doit donc être pertinente pour trouver une bonne correspondance, et le système doit être capable de remettre en question de mauvais éléments pour réaliser une remédiation de la reconnaissance. Pour la tâche de création de feedbacks, il faut s'assurer que la création de l'élève respecte les différentes règles algorithmiques de la pose d'opération. La réalisation de l'élève peut ne pas correspondre à ce qui est attendu : des tracés inattendus peuvent être présents ou à l'inverse manquants, il

est important que l'analyse puisse "rejeter" des tracés ne faisant pas partie directement de l'opération (des séparations entre unité et dizaines, des cercles supplémentaires...) afin de ne pas surgénérer des feedbacks erronés.

3.1.2 Architecture globale et préambule sur la suite du manuscrit

La figure 3.1 présente l'architecture générale du système d'analyse proposé. Deux entrées sont reçues, la consigne de l'exercice et les tracés de l'élève. Un exemple spécifique est présenté dans cette figure pour voir l'évolution du traitement des données. Pour des raisons de simplification, le procédé est présenté "a posteriori", c'est-à-dire une fois l'ensemble des tracés réalisés. En pratique, des stratégies permettant de traiter les tracés au cours de la saisie afin d'accélérer les différentes étapes du système seront proposées.

Les trois tâches décrites précédemment (reconnaissance, interprétation, feedbacks) sont réalisées par trois modules correspondants.

Module de reconnaissance. Le module de reconnaissance est présenté dans la Section 3.2. L'ensemble des tracés manuscrits sont utilisés en entrée du système pour produire une représentation de l'opération sous la forme d'un graphe (Section 3.2.1 à 3.2.4). Ce graphe permet de représenter la structure en deux dimensions de l'opération ; un nœud du graphe représente un symbole mathématique, et les liens entre nœuds représentent les relations spatiales mathématiques liant ces symboles. Afin d'éviter des erreurs de reconnaissances, le système est capable de générer plusieurs graphes hypothèses avec différentes segmentations (Section 3.2.2). Une procédure incrémentale est proposée afin d'éviter d'appliquer l'ensemble des traitements a posteriori de la saisie complète de manière à accélérer le traitement complet de l'opération (Section 3.2.5).

Module de mise en correspondance. Le module de mise en correspondance est présenté dans la Section 3.3. La pose d'une opération arithmétique respectant des contraintes algorithmiques, il est possible de générer une représentation de la solution attendue sous forme de graphe à partir de la consigne. Cette représentation est utilisée pour réaliser une mise en correspondance avec le graphe obtenu par le module de reconnaissance (Section 3.3.1) permettant de produire l'interprétation de la saisie. On propose une représentation intermédiaire sous forme de graphes de lignes permettant une accélération de la mise en correspondance (Section 3.3.3). Cette approche est étendue par un processus incrémental qui repose sur une génération progressive d'hypothèses et d'interprétations, afin

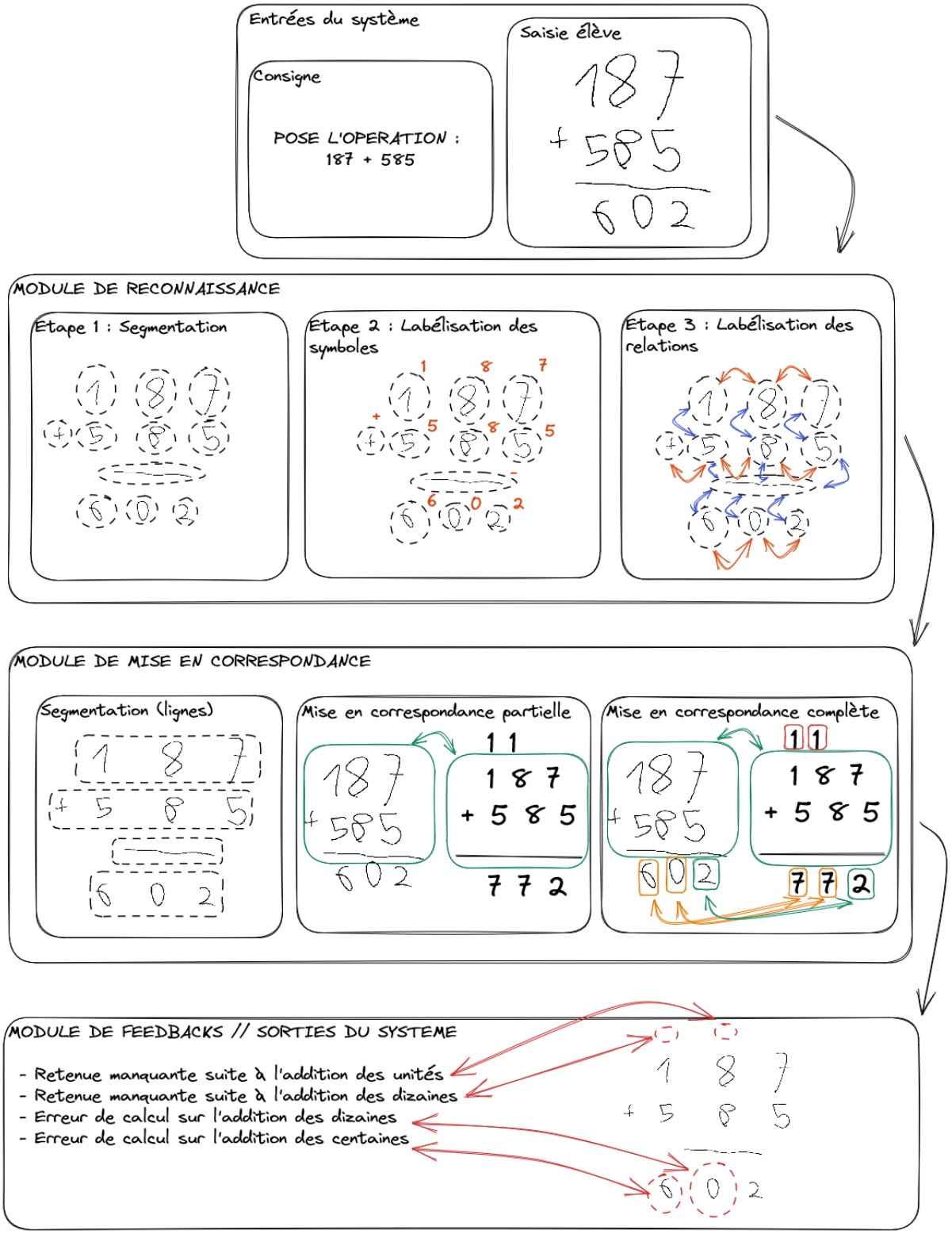

FIGURE 3.1 – Architecture du système d'analyse d'opérations arithmétiques. Les tracés de l'élève sont fournis au module de reconnaissance afin de produire une représentation sous forme de graphes. La consigne, transformée en graphe modèle, est fournie en parallèle du graphe représentant les tracés de l'élève obtenu par le module de reconnaissance pour obtenir une mise en correspondance des symboles. Cette mise en correspondance de graphes permet la génération de feedbacks.

d'optimiser l'exploration des hypothèses de graphes reconnues (Section 3.3.4).

Module de feedbacks. Le module de génération de feedbacks est présenté dans la Section 3.4. Une liste de symboles ne trouvant pas de correspondance, ayant une mauvaise correspondance, ou des symboles attendus n'étant pas présent serviront de base pour générer les feedbacks à l'élève. Une erreur sur le résultat pouvant être par exemple due à une erreur effective de calcul, une erreur dans la recopie des chiffres de la consigne ou à un oubli de retenue. Des règles sémantiques sont proposées pour transformer l'ensemble de correspondance en feedbacks intelligibles.

3.2 Module de reconnaissance : création de graphes d'opérations arithmétiques

Dans cette section, nous présentons plusieurs contributions sur la chaîne de traitement de reconnaissance d'opérations arithmétiques. Nous transformons un ensemble de tracés saisis par un élève en une représentation sous forme de graphes, en groupant les tracés en un ensemble de symboles et de relations mathématiques qui seront interprétés par la suite. Nous présentons le pré-traitement (Section des données 3.2.1) pour extraire les caractéristiques utilisées pour l'apprentissage des différents modèles de reconnaissance liés à la segmentation (Section 3.2.2), la labélisation des symboles (Section 3.2.3) et la labélisation des relations (Section 3.2.4). Afin d'aborder la problématique de traitement de l'information en un temps d'attente raisonnable pour l'élève, nous proposons une approche incrémentale (Section 3.2.5) qui permettra d'accélérer et d'éviter de réaliser l'intégralité des calculs pendant que l'opération est saisie par l'élève. Enfin nous proposons une approche pour corriger les erreurs de reconnaissance potentielles en générant de multiples hypothèses (Section 3.2.2).

La figure 3.2 reprend la chaîne de traitement de reconnaissance. Un ensemble de tracés représentant une opération complète ou partielle est fourni. Comme nous avons vu lors de la présentation concernant l'état de l'art de la reconnaissance d'opérations mathématiques, nous privilégions la séparation des différentes tâches de reconnaissance en sous-modules traitant chacune une tâche. Cette séparation permet notamment la construction itérative du module de reconnaissance au cours de la saisie, mais permet aussi de proposer à chaque niveau certaines hypothèses de reconnaissance qui peuvent être retravaillées sans traiter l'entièreté de la chaîne. L'objectif de la tâche de segmentation est de créer une première représentation sous la forme d'un graphe, chaque nœud regroupant un ou plusieurs tracés

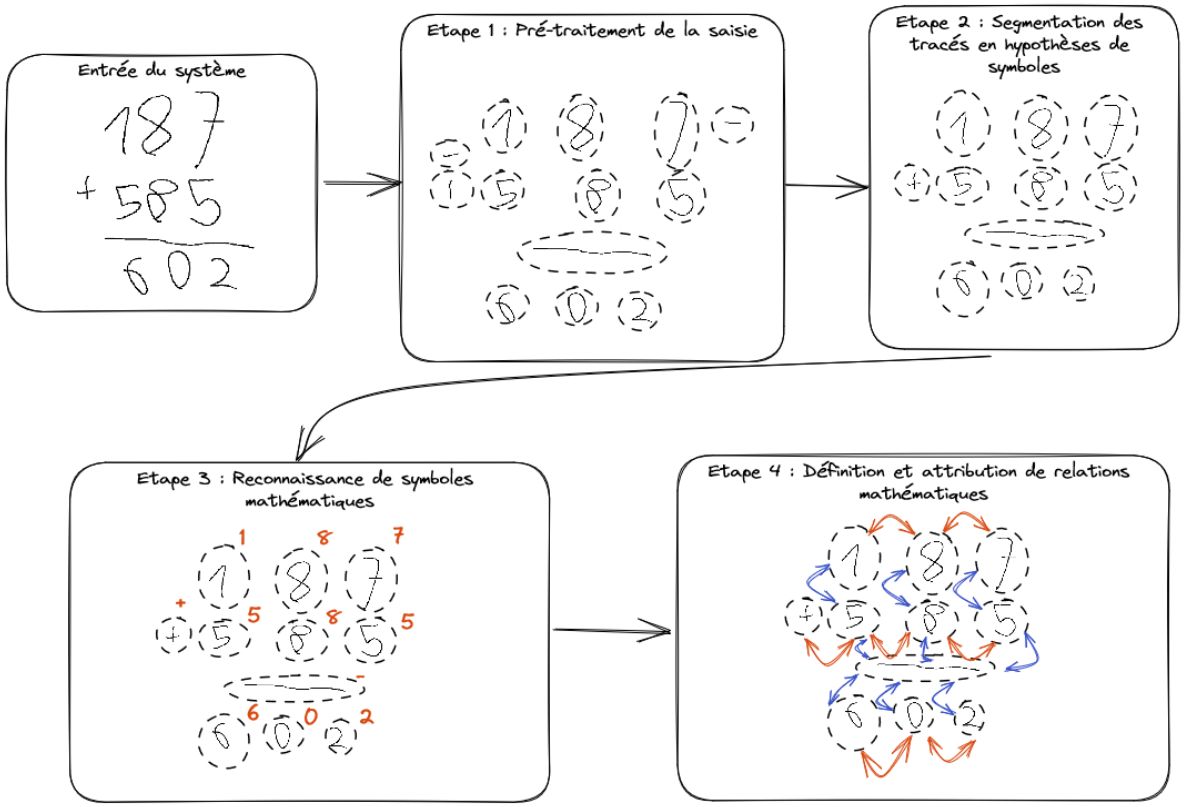

FIGURE 3.2 – Chaîne de traitement du module de reconnaissance. L'ensemble des tracés est d'abord pré-traité dans une première étape (Section 3.2.1), un premier graphe est créé pour extraire des caractéristiques de paires de tracés afin de regrouper les tracés en nœud unique représentant un symbole (Section 3.2.2). Ensuite un label mathématique est fixé pour chaque nœud (Section 3.2.3) puis des relations spatiales entre symboles sont évaluées (Section 3.2.4).

représentant un symbole et des liens représentant de potentielles relations mathématiques qui seront traitées par la suite.

3.2.1 Étape 1 : Pré-traitement du signal "en-ligne" des tracés manuscrits

Afin de limiter l'impact de la variabilité des saisies sur les tâches de reconnaissance, il est classique d'appliquer un ensemble de pré-traitements avant de procéder à l'apprentissage et l'application des modèles. Cela permet d'uniformiser ces données pour les rendre moins dépendantes des conditions de saisies. Les saisies manuscrites peuvent en effet être très hétérogènes, en fonction :

— des différentes tablettes et stylets (fréquences d'échantillonnage) ;
— des vitesses de saisies entre élèves ;
— des hésitations de tracés pouvant être introduits suivant les élèves.

Suppression des points superposés. La première étape consiste à supprimer les points temporellement adjacents qui se superposent entre eux.

Lissage du tracé. Suivant l'expertise de l'utilisateur avec la tablette et le stylet, mais aussi l'âge de l'élève, les tracés peuvent se retrouver déformés, on peut constater des tremblements. Afin d'en limiter l'impact, une étape de lissage est réalisée.

Normalisation des tracés. Les tracés sont normalisés en calculant les minimas et maximas des ordonnées afin que les valeurs soient comprises dans l'intervalle [0, 1] et en adaptant les dimensions sur les abscisses pour ne pas déformer les tracés.

Re-échantillonnage des tracés. La fréquence d'échantillonage peut être très variable et parfois atteindre plusieurs centaines de points par tracé. Dans le cadre de caractéristiques extraites entre paires de points pour chaque paire de tracés, la quantité d'opérations à réaliser croit de manière exponentielle. On peut donc réduire et normaliser l'échantillonnage. L'impact de ce ré-échantillonnage sur les différents processus de reconnaissance et sur les temps de calculs sera évalué par la suite.

Interpolation. À l'inverse, il est possible qu'un tracé ne contienne pas suffisamment de points dû à des saisies très rapides ou un échantillonnage trop faible de la tablette. Afin d'avoir une répartition uniforme des points, nous pourrons appliquer cette interpolation au cas par cas en fonction des étapes de traitement, notamment pour réduire la combinatoire dans le cadre de l'extraction de certaines caractéristiques.

La figure 3.3 présente deux exemples de tracés sur lesquels sont appliquées le prétraitement. Plusieurs taux d'échantillonnage sont mis en avant afin d'observer la conservation ou non de la forme du tracé en fonction des valeurs définies. Pour l'extraction de caractéristiques géométriques, 50 points par tracés seront conservés, tandis que 15 points sont utilisés pour les calculs d'angles de vision.

3.2.2 Étape 2 : Segmentation des tracés en hypothèses de symboles

Initialisation du graphe de segmentation

L'étape d'initialisation du graphe de segmentation permet de définir une stratégie pour considérer les paires de tracés qui peuvent potentiellement appartenir au même symbole.

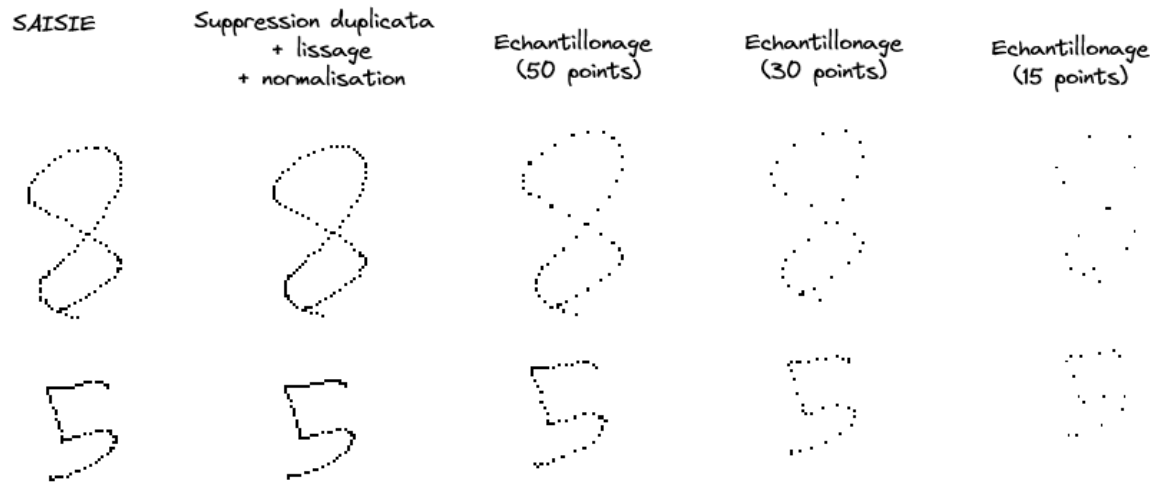

FIGURE 3.3 – Pré-traitement des tracés appliqués pour homogénéiser les saisies et d'harmoniser ou réduire la quantité de points pour l'extraction de caractéristiques.

Il n'est pas envisageable de faire la combinatoire de toutes les paires de tracés, notamment sur de grandes opérations. Plusieurs stratégies ont été proposées dans l'état de l'art, en créant des liens entre les plus proches voisins dans l'espace de saisie ou par leur position temporelle. Des ré-agencements temporels sont proposés pour anticiper des va-et-vient lors de la saisie de l'opération. Cependant ces approches peuvent éliminer de nombreuses hypothèses intéressantes si l'on restreint trop les possibilités de liens entre chaque tracé, ou peuvent conserver un trop grand nombre d'hypothèses si le nombre de voisins choisis est trop important. Nous avons choisi de nous inspirer de l'approche proposée par [HZ16a] qui se base sur la visibilité entre tracés, pour limiter la quantité de paires de tracés considérées tout en conservant l'intégralité des paires pertinentes.

Pour chaque tracé, des *angles de visions* sont calculés entre son centre et l'ensemble des autres tracés de l'opération, du plus proche tracé au plus éloigné. Si un tracé est considéré "visible", c'est-à-dire que l'angle de vision du tracé n'est pas masqué par des tracés plus proches, alors un lien logique est créé entre la paire de tracé. Les angles de vision vers ce tracé sont ensuite considérés obstrués pour les tracés plus éloignés. Pour gagner du temps de calcul, les angles de vision ne sont pas calculés sur l'ensemble des segments composant un tracé mais sur les boîtes englobantes de ces derniers. La figure 3.4 présente cette recherche et validation d'angles de vision dans l'espace pour un tracé donné. C'est à cette étape que nous appliquons par ailleurs le pré-traitement de ré-échantillonage sur les tracés. En effet les calculs d'angles de vision ne sont pas affectés par la réduction du

nombre de points des tracés puisque les formes sont conservées, et cela évite de réaliser ce calcul sur une quantité trop importantes de segments de tracés. Cette étape d'initialisation permet d'aboutir au graphe de segmentation présent sur cette même figure, où un nœud correspond à un tracé unique dans l'opération et un lien correspond à une paire qui sera considérée pour la segmentation.

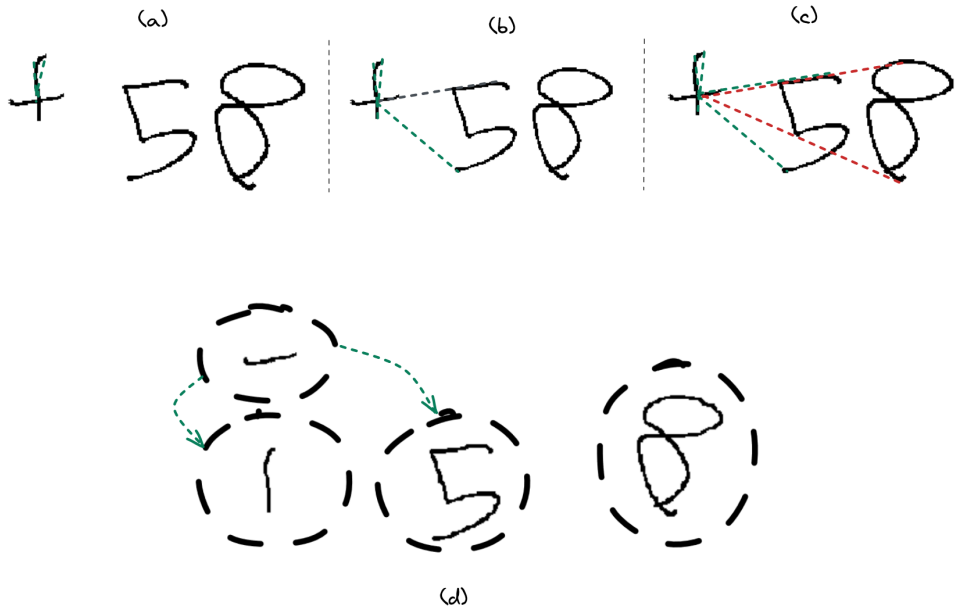

FIGURE 3.4 – Étape d'initialisation du graphe. Pour chaque paire de tracés, des angles de vision sont calculés pour déterminer la pertinence d'évaluer les paires de tracés considérées. (a) Un premier angle de vision est calculé entre le tracé horizontal du + et le tracé vertical du +, puis (b) un nouvel angle de vision entre le tracé horizontal du + et le 5. Enfin (c) le dernier angle de vision entre le tracé horizontal du + et le 8 est complètement masqué par le 5. On obtient (d) un premier graphe qui contiendra des liens qui identifient les paires de tracés qui seront considérées pour l'étape suivante. Cette opération est répétée pour l'ensemble des tracés de l'opération.

Extraction de caractéristiques pour la segmentation de tracés

L'objectif est maintenant d'extraire un ensemble de caractéristiques pour déterminer si des paires de tracés font partie d'un même symbole. Nous reprenons les caractéristiques proposées par [HZ16a] et présentées dans la Section 2.2.2 et les étendons avec un ensemble de caractéristiques géométriques. Pour le calcul des caractéristiques ci-dessous, on note P_1T_1 et P_1T_2 le premier point du premier et du second tracé, P_nT_1 et P_nT_2 le dernier

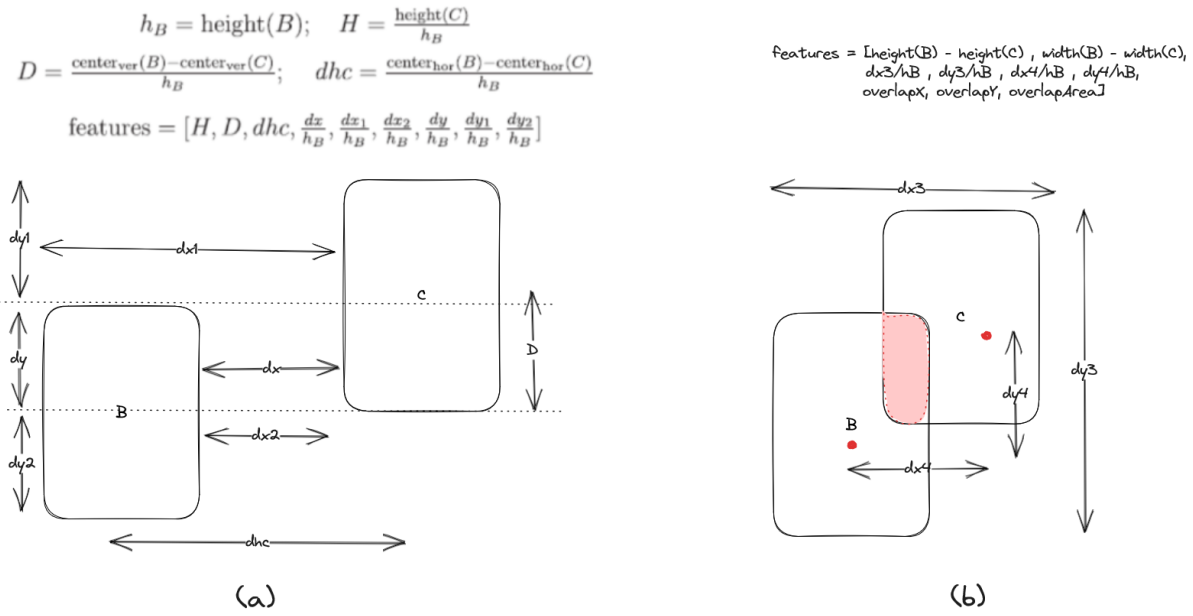

FIGURE 3.5 – Ensemble des caractéristiques géométriques extrait sur les boîtes englobantes et centres des tracés. (a) Ensemble présenté dans [ASB14], (b) Ensemble complémentaire.

point du premier et du second tracé, $P_C T_1$ et $P_C T_2$ les centres des deux tracés, $P_{CM} T_1$ et $P_{CM} T_2$ les centres de masse des deux tracés :

- 120 caractéristiques de contexte de forme ([HZ16a])
- 9 caractéristiques présentées dans [ASB14] (figure 3.5 (a))
- 9 caractéristiques complémentaires (figure 3.5 (b)) basées sur les boîtes englobantes, centre des tracés, et sur la zone de chevauchement des tracés
- 8 caractéristiques basées sur les proximités entre les points d'intérêts des tracés : $[P_{1_x} T_1 - P_{1_x} T_2,\ P_{1_y} T_1 - P_{1_y} T_2,\ P_{1_n} T_1 - P_{1_n} T_2,\ P_{n_y} T_1 - P_{n_y} T_2,\ ED(P_1 T_1 - P_1 T_2),\ ED(P_n T_1 - P_n T_2),\ P_{n_x} T_1 - P_{1_x} T_2,\ P_{n_y} T_1 - P_{1_y} T_2]$, où ED représente le calcul de la distance euclidéenne entre deux points.
- 6 caractéristiques basées sur des angles entre les points d'intérêts des tracés et l'origine : $[a(P_1 T_1, P_1 T_1),\ a(P_1 T_1, P_1 T_2),\ a(P_n T_1, P_1 T_2),\ a(P_n T_1, P_n T_2),\ a(P_C T_1, P_C T_2),\ a(P_{CM} T_1, P_{CM} T_2)]$.
- 1 caractéristique représentant la temporalité t observée entre la saisie de la paire de tracés.

Pour rester indépendant des différentes tailles de tracés au sein d'une même opération, les caractéristiques basées sur des distances sont normalisées en fonction de la hauteur

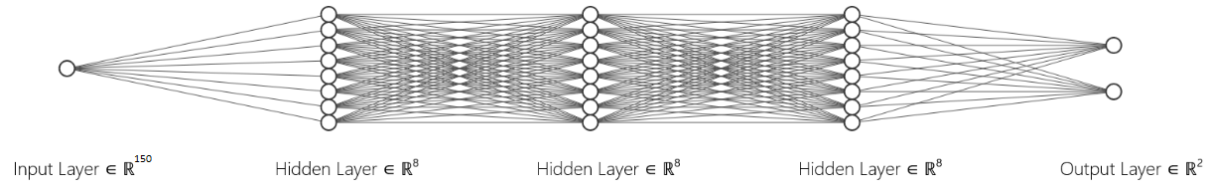

FIGURE 3.6 – Modèle Fully-Connected proposé pour réalisé la tâche de segmentation. 150 caractéristiques sont fournies en entrée du système pour déterminer l'appartenance ou non d'une paire de tracés à un unique symbole. Le modèle se compose de 3 couches FC.

moyenne des tracés de l'opération. Pour éviter tout biais de tracés particulièrement petits (barre horizontale) ou grands (séparateur vertical), ces derniers sont ignorés dans le calcul de la moyenne.

Modèles de segmentation : classification en symbole / non symbole

Un classifieur binaire est appris pour déterminer à partir des caractéristiques extraites si une paire de tracés doit être fusionnée en un symbole unique. De nombreux classifieurs peuvent être utilisés (SVM, forêts d'arbres décisionnelles, réseaux de neurones..). Nous proposons d'utiliser pour cette tâche un simple modèle de réseau de neurones profond de type "Fully-Connected" (voir figure 3.6) qui a l'avantage d'être assez léger et rapide pour être embarqué sur application mobile. On dispose en entrée d'un vecteur de 153 caractéristiques (120 de formes, 33 supplémentaires), 3 couches FC (de taille 8) sont enchaînées et enfin une fonction softmax permet de définir la classe de la paire de tracés ("même symbole" / "non symbole"). Cette architecture permet d'avoir un modèle suffisamment petit pour que le temps d'exécution soit faible tout en fournissant de bons scores de reconnaissance.

Évolution du graphe suite à la segmentation

Chaque paire de tracés ayant un lien dans le graphe de segmentation est donc classifiée soit en "même symbole" soit en "non symbole" par notre réseau de neurones. Une nouvelle représentation sous forme de graphe est donc définie. Pour chaque paire de nœuds du graphe de segmentation, si au moins un des liens est considéré comme un même symbole, alors un nœud est créé. Les liens entre les nouveaux nœuds constitués sont hérités des liens entre tracés existants et seront ensuité traités pour déterminer si ces derniers représentent

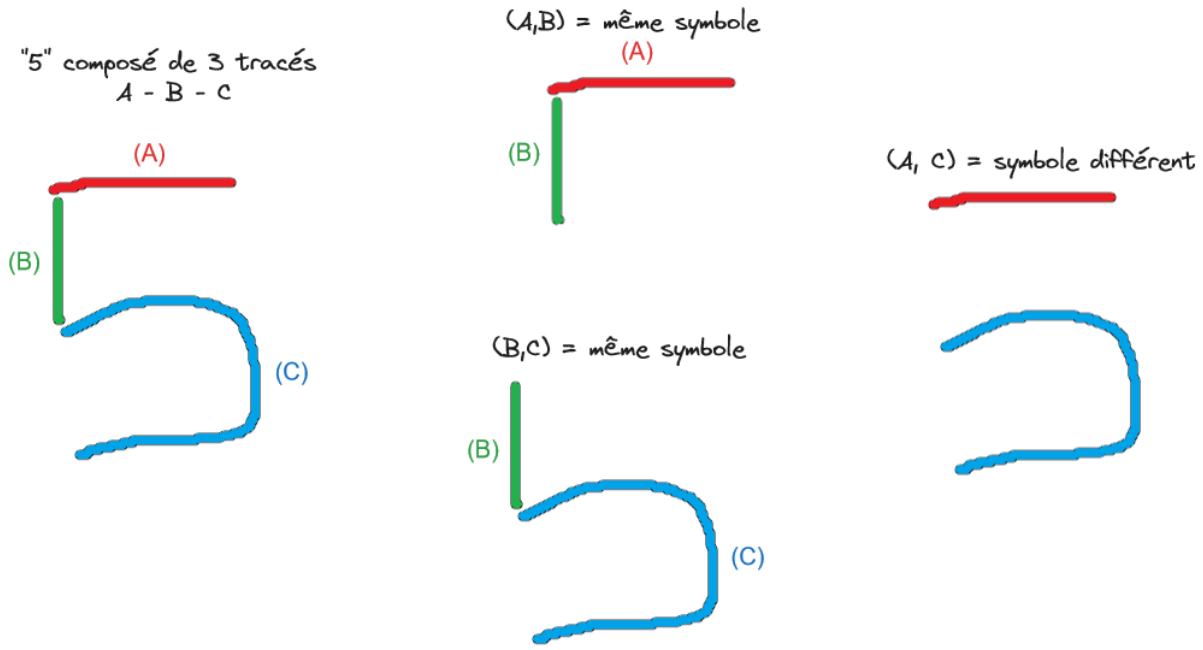

FIGURE 3.7 – Exemple d'un 5 composé de 3 différents tracés. Les paires de tracés (A,B) et (B,C) sont considérés comme étant du même symbole, mais pas la paire de tracé (A,C). Par transition, cette paire est considérée au sein du même symbole.

des alignements mathématiques au sein de l'opération. Un symbole pourvant être composé d'une multitude de tracés (deux ou plus) pouvant être disjoint entre paire de tracés, comme c'est le cas du symbole 5 sur la figure 3.7, alors le processus de création de nœud est transitif : si deux paires de tracés (A,B) et (B,C) sont toutes les deux considérées comme un même symbole, alors ces trois tracés formeront un uniquement symbole (A,B,C) même si la paire (A,C) n'est pas considérée comme appartenant au même symbole, comme cela serait le cas pour la barre horizontale avec la boucle d'un symbole 5 à cause de leur éloignement spatial.

Bilan du traitement des tracés

La figure 3.8 représente ce nouveau graphe. Dans cette section, nous avons traité la seconde étape du module de reconnaissance, à savoir l'extraction des caractéristiques permettant la segmentation de tracés en un ensemble d'hypothèses de symboles. L'ensemble de tracés est transformé en un premier graphe dont chaque nœud représente une hypothèse de symbole mathématique, et chaque lien au sein du graphe représente une relation spatiale qui sera définie par la suite.

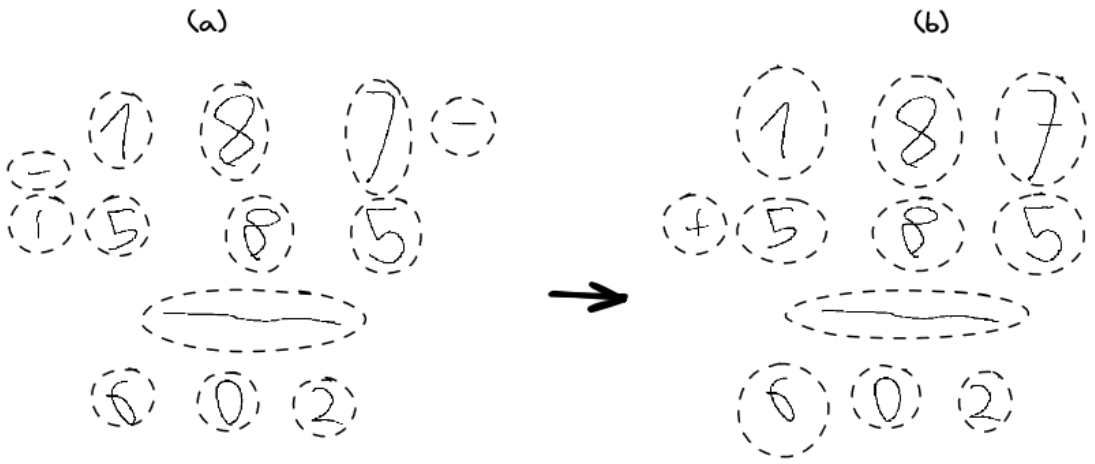

FIGURE 3.8 – Transformation du graphe de tracés (a) en un premier graphe de symboles (b). Les tracés étant déterminés comme fusionnés constituent un seul nœud. Les liens entre les nouveaux nœuds sont hérités du précédent graphe.

Formalisation de l'ensemble des hypothèses de graphes

Ce premier regroupement de tracés en symboles obtenu à la suite des étapes décrites précédemment n'est pas toujours satisfaisant pour aboutir aux résultats attendus. Les tracés d'un élève peuvent être trop séparés, ou au contraire différents symboles peuvent se superposer suite à des corrections ou des ajouts comme des retenues ou des reports qui viennent se superposer à des tracés existants. Les erreurs de regroupement de tracés entraînent par la suite des erreurs sur les classifications de symboles ou sur les positionnements des symboles entre eux. Ces erreurs conduiraient alors à une interprétation incorrecte des tracés de l'élève, et donc à un mauvais feedback. Afin d'éviter de produire en amont ces erreurs, nous proposons une stratégie de génération de plusieurs hypothèses. Par la suite le système va explorer itérativement ces hypothèses pour déterminer la meilleure en prenant en compte le contexte global dans l'analyse de l'opération.

Pour générer plusieurs hypothèses de segmentation, nous allons nous appuyer sur le score du classifieur de regroupement (fusion / non fusion) de chaque paire de tracés considérée. Ce score est utilisé pour déterminer si deux nœuds du graphe doivent être fusionnés. Si $s > 0.5$ alors un nouveau nœud est constitué. Si le score se situe entre $0.5 - \tau < s < 0.5 + \tau$, deux hypothèses de graphes sont créés, représentant d'un côté la séparation et de l'autre la fusion des tracés. Si $s < 0.5 - \tau$ alors les tracés restent séparés et si $s > 0.5 - \tau$, les tracés sont fusionnés sur l'intégralité des hypothèses. Le seuil τ peut être défini dynamiquement au cours de l'opération permettant d'éviter de générer

un nombre trop conséquent d'hypothèses pour des opérations où de nombreux symboles sont rapprochés ou superposés. Une stratégie pour manipuler ce seuil afin d'éviter la création de trop nombreuses hypothèses, tout en évitant d'omettre des interprétations intéressantes sera proposée en Section 3.3.4.

La figure 3.9 présente l'exemple d'une opération où de multiples hypothèses de segmentation différentes peuvent être produites. Le symbole 9, découpé en deux tracés, peut être considéré comme deux symboles. Le symbole 5 et les deux retenues superposées peuvent être considérés comme étant trois différents symboles, un seul symbole ou une combinaison de deux symboles.

3.2.3 Étape 3 : Reconnaissance de symboles mathématiques

Dans cette section, nous traitons la deuxième étape de notre chaîne de reconnaissance, à savoir la reconnaissance de symboles mathématiques. Nous avons obtenu après la seconde étape plusieurs graphes représentant chacun des hypothèses de segmentation de l'opération arithmétique, dont les nœuds représentent des ensembles de tracés pouvant correspondre à un symbole mathématique, et des liens entre ces nœuds représentant de possibles relations mathématiques entre symboles. Chaque graphe est constitué lorsqu'une hypothèse de segmentation dépasse un seuil, afin de ne pas rejeter une interprétation de l'opération pouvant être correcte. Pour l'instant, ni les nœuds ni les liens ne contiennent une information quant à leur nature. L'objectif de cette troisième étape est d'attribuer un label à chaque nœud permettant d'identifier la nature du symbole mathématique afin d'enrichir le graphe.

Modèle de reconnaissance et entrée du réseau

Pour l'attribution d'un label sur un ensemble de tracés, nous avons décidé de reprendre l'architecture de réseaux de neurones profonds VGG 16 [SZ14] communément utilisée pour de la reconnaissance de caractères hors-ligne. Cette architecture (figure 3.10) est composée de plusieurs couches de convolution et de pooling permettant l'extraction et la sélection de caractéristiques, puis une fonction soft-max est utilisée pour déterminer la classe du symbole associé.

Il est communément fourni en entrée une matrice représentant l'image (hors-ligne) que l'on souhaite faire reconnaître au système. Afin de profiter de la richesse de l'information du signal en-ligne dont nous disposons, nous étendons cette simple entrée en fournissant

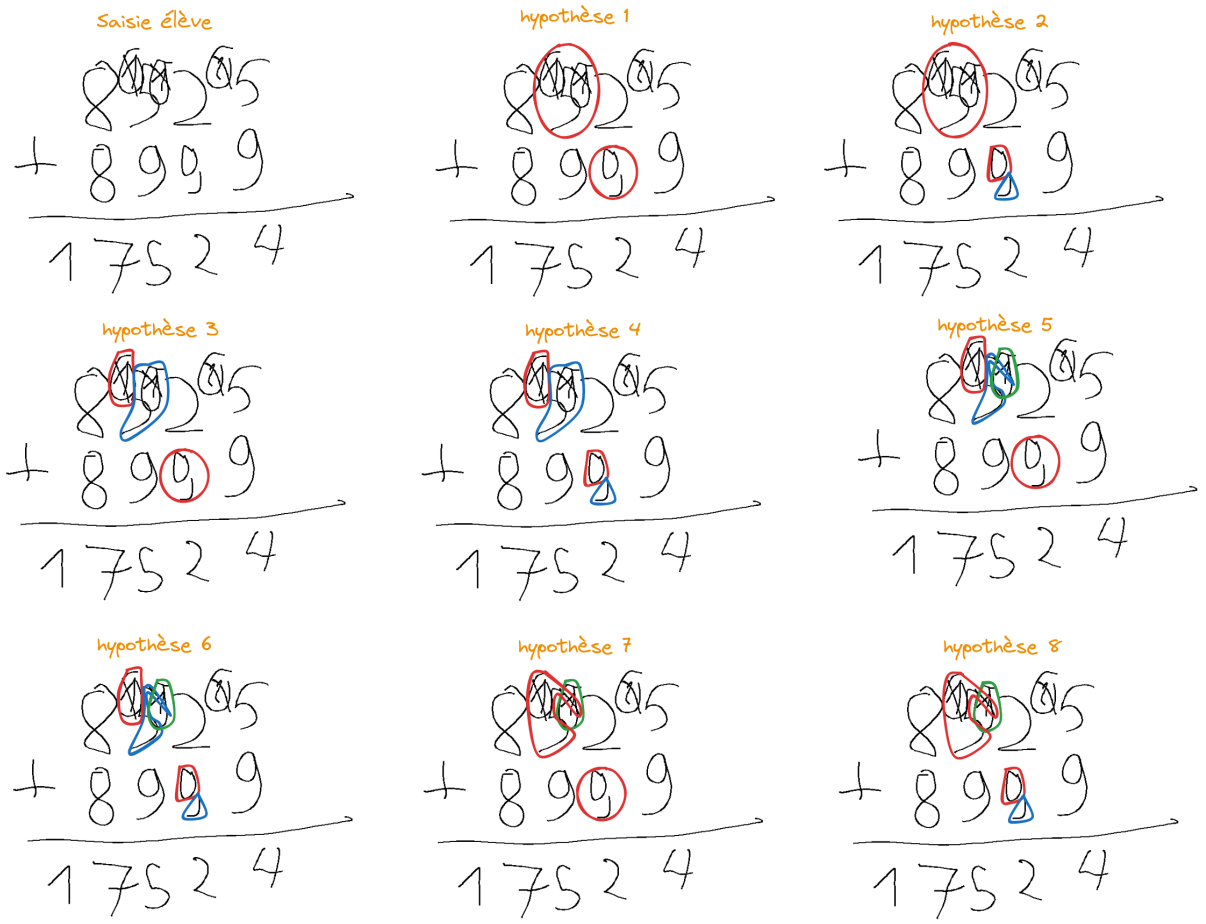

FIGURE 3.9 – Exemple d'une opération avec plusieurs hypothèses de segmentation. Pour un soucis de clarté, seul les nœuds variants entre hypothèses sont mis en évidence. Les tracés du symbole 9 peuvent être considéré comme un même tracé ou deux symboles disjoints. Les tracés du symbole 5 et des retenues superposées ont de multiples interprétation, allant d'un symbole unique à trois symboles disjoints, pour un total de 8 différents graphes hypothèses générés.

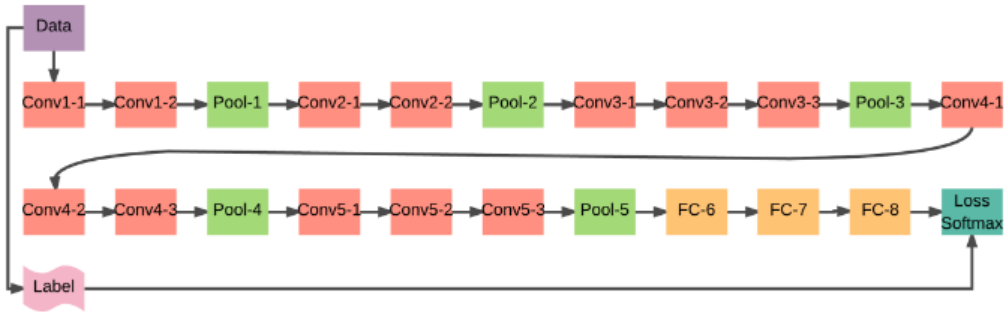

FIGURE 3.10 – Architecture de VGG16.

3 différents canaux au modèle. Chaque tracé est converti en trois matrices de 32 par 32 pixels qui représentent l'ensemble des points le constituant dans le plan. La première matrice représente l'image du tracé transformé du signal en-ligne en hors-ligne. Lorsque le segment entre deux points consécutifs du tracé passent par des cellules de la matrice, la valeur de ces dernières est fixée à 1. Les deux autres matrices représentent des directions de trajectoire en calculant l'angle formé par le tracé relativement à un vecteur horizontal : la matrice 2 est représentée par le cosinus et la matrice 3 par le sinus. Si aucun tracé ne passe par le pixel en question, alors la valeur sera de 0 (aucun mouvement sur ce pixel). Ainsi la matrice 2 représente le mouvement horizontal du tracé tandis que la matrice 3 représente le mouvement vertical. Si un tracé repasse plusieurs fois par le même pixel, alors la direction calculée représentera le dernier passage effectué. En fournissant ces 3 matrices, le modèle pourra extraire des caractéristiques sur les mouvements, permettant de discriminer certains symboles se ressemblant visuellement mais ayant un ordre d'écriture dissocié (les symboles 4 et + par exemple).

Bilan du traitement des symboles

La figure 3.12 représente ce nouveau graphe. Le graphe est maintenant enrichi par des labels pour chaque nœud représentant un symbole mathématique. Les symboles sont transformés en un signal sur 3 matrices représentant à la fois le signal hors-ligne du tracé du symbole, ainsi que les données en ligne de ce dernier en conservant les directions sur les deux axes des tracés. Le modèle de réseaux de neurones profonds basé sur l'architecture VGG16 est suffisamment peu coûteux en mémoire mais avec des résultats efficaces pour réaliser la tâche de reconnaissance de symboles mathématiques.

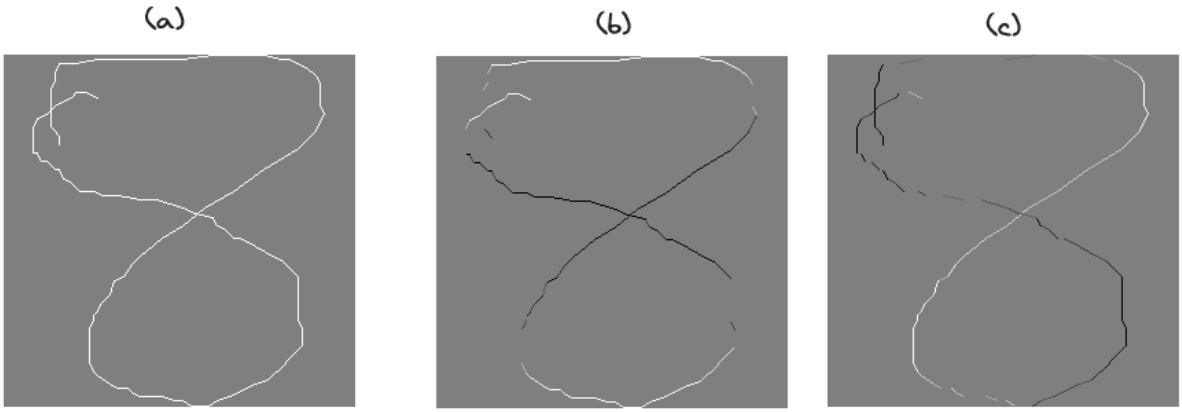

FIGURE 3.11 – Représentation visuelle des trois matrices d'un symbole. (a) représente le tracé brut de l'opération, (b) le cosinus de l'angle formé par le tracé avec l'axe horizontal et (c) le sinus de l'angle formé par le tracé avec l'axe horizontal.

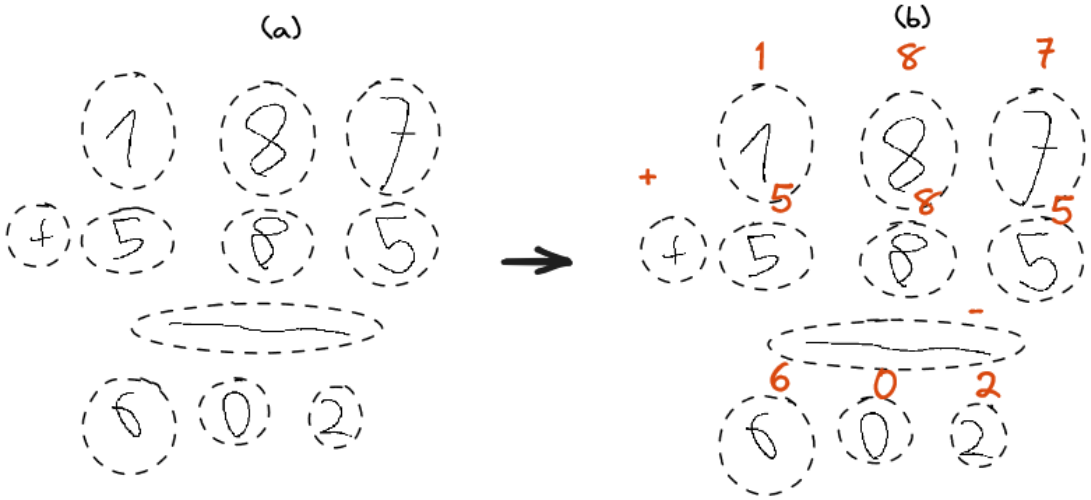

FIGURE 3.12 – État du graphe suite à l'attribution des labels. Chaque nœud est un regroupement de tracés manuscrits portant le label du symbole mathématique reconnu.

3.2.4 Étape 4 : Définition et attribution de relations mathématiques

Dans cette section, nous traitons la dernière étape de la chaîne de reconnaissance. Nous avons en entrée un ensemble de graphes hypothèses représentant l'opération arithmétique saisie. Les nœuds possèdent maintenant un label représentant la nature des symboles reconnus. Les liens entre nœuds représentent les potentielles relations mathématiques, sans labels actuellement, entre symboles. L'objectif est d'attribuer et d'évaluer la qualité de chaque relation, et de supprimer les liens qui ne représentent aucune relation. Nous obtiendrons ainsi un ensemble complet de graphes représentant les symboles et leurs relations au sein de l'opération afin de pouvoir procéder alors à l'interprétation de la saisie de l'élève étant donnée une consigne.

Les paysages flous mathématiques. Il y a de nombreuses caractéristiques qui peuvent être utilisées pour l'étape de reconnaissance des relations mathématiques. Elles se basent sur des ensembles de distances et d'angles entre symboles assez discriminants pour qualifier les ensembles disjoints <droite> <gauche> <haut> <bas>. Cependant dans le plan en deux dimensions d'une opération posée à main levée par un élève, les relations spatiales entre les éléments de l'opération arithmétique ne sont pas simples à qualifier.

De plus notre objectif n'est pas simplement de reconnaître ce que l'élève a fait mais aussi de qualifier la réalisation. Pour cela, il est nécessaire de disposer d'une caractérisation plus fine des relations spatiales pour produire des feedbacks les plus pertinents possibles.

Nous proposons d'utiliser la notion de "paysage flou" [Blo99] permettant de représenter et de qualifier des relations spatiales. Le principe consiste à apprendre, à partir d'un ensemble de zones angulaires et de distances inter-éléments, la représentation d'une relation. Cette représentation se base à la fois sur la morphologie des symboles "parent" et "enfant" ainsi que sur leur position relative. L'objectif est de délimiter une zones adéquate qui correspond à la relation modélisée. En combinant un ensemble de paires de différents symboles respectant la position attendue, il est possible d'apprendre un paysage flou comme présenté dans [DA14], permettant d'attribuer à chaque zone de l'espace une valeur qualitative de la relation. La figure 3.13 représente l'évolution d'un paysage flou directionnel vers un paysage flou mathématique appris, résultant de la combinaison des quatres paysages flous directionnels (à droite de, à gauche de, en haut de, en bas de) ainsi que d'un paysage flou de distance.

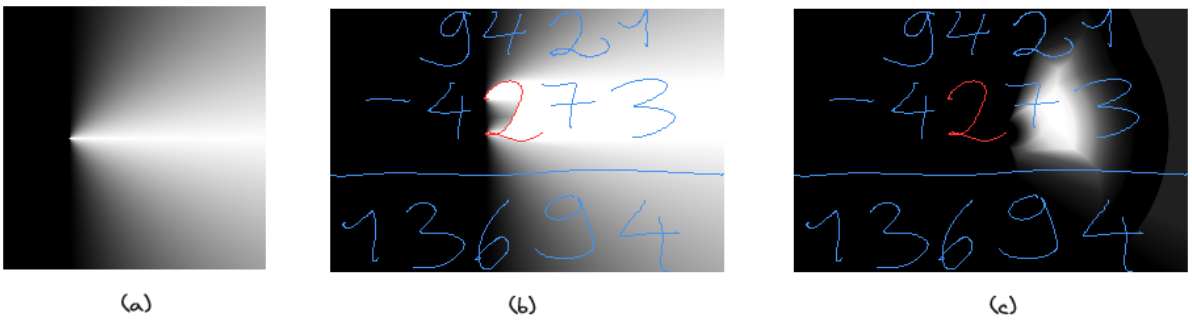

FIGURE 3.13 – Représentation de l'évolution des paysages flous. Plus un pixel est blanc, plus la valeur d'appartenance au paysage est forte (a) paysage mono-directionnel "à droite de.." à partir d'un point, (b) paysage mono-directionnel "à droite de.." à partir d'un (rouge), (c) paysage multi-directionnel avec distance appris pour la relation mathématique Droite."

Extraction de paysages adaptés aux opérations arithmétiques. Il est nécessaire de bien définir les paysages flous nécessaires aux opérations arithmétiques afin de bien représenter les relations. Nous identifions 9 relations mathématiques disjointes : Droite, Gauche, Haut, Bas, Retenue gauche, Retenue droite, Report gauche, Haut barre, Bas barre. Notons que la relation Haut/Bas entre un chiffre et une barre d'opération est différenciée de la relation Haut/Bas entre une paire de chiffres, cette même relation mathématique correspondant à des zones différentes de l'espace de recherche relativement au corps des symboles observés.

Caractéristiques géométriques. Les paysages présentés permettent de définir des zones précises relatives aux symboles afin d'en déterminer la relation. Afin d'affiner l'espace de recherche, nous proposons d'utiliser d'autres caractéristiques géométriques pour compléter l'espace vectoriel permettant de déterminer la relation. Les caractéristiques à contexte de forme présentées en Section 3.2.2 permettront avec les valeurs des paysages flous de déterminer précisément la position des symboles les uns par rapport aux autres.

Modèle d'apprentissage. Nous utilisons la même architecture de classifieur utilisé pour la segmentation (figure 3.6) pour l'apprentissage des relations mathématiques. Cette fois-ci le classifieur n'est pas binaire mais présente en sortie un vecteur de dimension 9 pour l'identification des relations.

Attribution des labels et graphes finaux. Pour renforcer la notion de visibilité, on utilise les symboles précédemment traités pour masquer l'espace de recherche. Ainsi pour chaque symbole, on va chercher à attribuer une relation aux symboles voisins, du

plus proche au plus éloigné de l'espace. Pour une paire de symboles (S1, S2), si S2 est déterminé comme ayant une relation DROITE avec le symbole S1, il masquera par la suite le paysage associé à la relation mathématique DROITE pour tous les autres symboles plus éloignés de l'opération dont la relation avec S1 est étudiée. Dans [DA14], la formule 3.1 attribue un degré d'appartenance en calculant l'angle entre un vecteur direction donné $\vec{\mu_\alpha}$ avec le vecteur $\vec{op}$ qui représente le vecteur entre le point de référence et le point évalué. α est la nature de la direction considérée (à droite de, en haut de...) qui va définir le vecteur.

$$v_\alpha(p) = max(0, 1 - \frac{2}{\pi}.arccos(\frac{\vec{op}.\vec{\mu_\alpha}}{\vec{op}})) \tag{3.1}$$

On propose pour prendre en compte les angles de visions masqués de faire intervenir f, qui représente la valeur de masquage du paysage pour les tracés étudiés. f a une valeur m pour une direction donnée si le vecteur $\vec{op}$ se trouve dans l'ensemble masqué O, sinon 0. Cela permet de prendre en compte des opérations dont les symboles sont très rapprochés, afin d'éviter de relier par une relation mathématique deux symboles si d'autres symboles sont observés plus proches pour la même relation.

$$v_\alpha(p) = max(0, 1 - \frac{2}{\pi}.arccos(\frac{\vec{op}.\vec{\mu_\alpha}}{\vec{op}}) - f), \forall p \in S \tag{3.2}$$

$$f = \begin{cases} m & if \vec{op} \in O \\ 0 & else \end{cases} \tag{3.3}$$

Bilan du traitement des relations

Dans cette section, nous avons vu l'apprentissage de paysages flous dédiés aux opérations arithmétiques pour représenter des relations spatiales, à partir de la morphologie des symboles et de leur position relative. Ces paysages permettent de représenter précisément des zones qui correspondent aux attentes de positionnement de symboles étant donnée une relation mathématique. Cela permet de créer un classifieur pour discriminer ces relations, mais aussi d'identifier précisément les zones de saisies intéressantes pour la création des feedbacks pour l'élève, à la fois pour lui indiquer les zones en erreur, ou pour le guider dans ses saisies.

De plus, ces paysages flous permettent d'évaluer qualitativement la position du tracé. Ces valeurs d'appartenance à chaque paysage mesurent la correspondance entre deux symboles vis-à-vis de la relation interprétée. Ces dernières informations seront formalisées

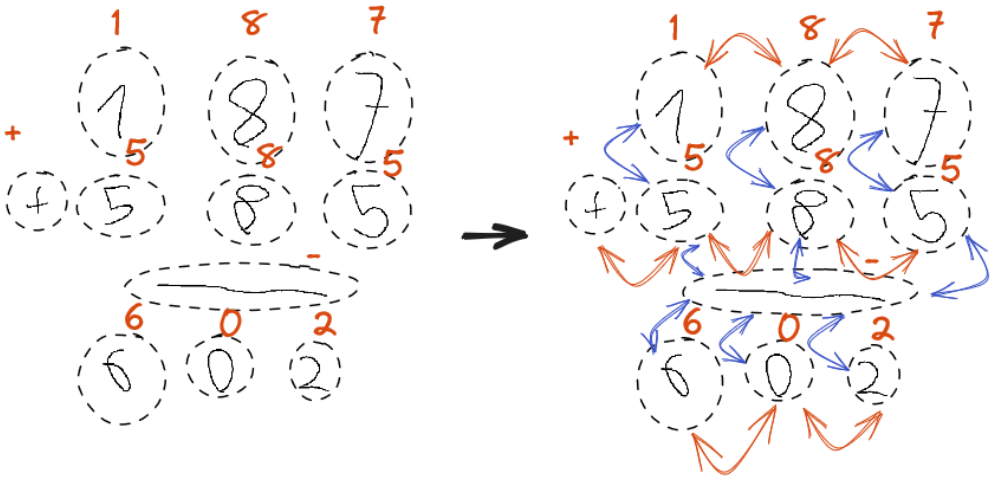

FIGURE 3.14 – État de l'ensemble de graphes de reconnaissance suite à l'attribution des labels pour chaque relation entre symboles. Chaque lien entre nœuds représente une relation mathématique reconnue.

sur les liens du graphe et seront utilisées dans l'interprétation des tracés par la suite de la chaîne de traitement. Les hypothèses de segmentation, de reconnaissance de symboles et de relations spatiales sont formalisées par un ensemble de graphes qui seront à la base des interprétations des saisies de l'enfant. La figure 3.14 représente le passage du graphe avec uniquement les nœuds annotés au graphe contenant les liens annotés.

3.2.5 Version incrémentale de la création de graphes de segmentation

Processus incrémental de segmentation. Nous présentons dans cette section différentes optimisations permettant de réduire les temps de calcul. Afin de tirer profit du contexte de saisie, l'écriture et la réalisation d'un exercice en temps réel par l'enfant, nous proposons d'effectuer l'étape de segmentation, et ainsi les étapes sous-jacentes présentées auparavant, de manière incrémentale. L'objectif est de ne pas devoir calculer la segmentation sur l'ensemble de l'opération à la fin de la saisie, ni de reproduire des calculs précédemment considérés sur certains tracés.

Les tracés sont traités par groupe. La première segmentation s'effectue sur un premier ensemble de tracés. Les distances entre tracés et les angles de vision sont conservés pour chaque tracé. Lorsqu'un nouveau tracé est ajouté pour traitement, on reproduit l'étape d'initialisation du graphe avec les précédents angles de vision calculés. Ainsi pour chaque

tracé précédemment observé, on recherche un angle de vision sur chaque nouveau tracé uniquement, et pour chaque nouveau tracé on calcule les angles de vision avec le reste de l'opération. On ne remet pas en cause les précédents liens, même si un nouveau tracé intervient et masquerait une ancienne paire de tracés. Ainsi, considérant n la quantité de tracés de la saisie complète et m la quantité de tracés au cours de la saisie, on calcule au cours de la saisie $2 * (m - 1)$ nouvelles potentielles paires de segmentation, au lieu de $(n - 1)^2$ au terme de la saisie.

Processus incrémental d'attribution de relations Pour identifier les symboles, il suffit d'appliquer la classification (cf. Section 3.2.3) sur les nouveaux nœuds et liens sans labels des graphes suite à l'ajout de chaque tracé. Si la segmentation d'un précédent nœud est remise en question, alors les relations précédemment établies sont ré-évaluées.

On reprend donc l'algorithme de segmentation, mais cette fois-ci en héritant d'abord des relations des tracés nouvellement segmentés. Puis, les paysages sont à nouveau calculés à partir du nouveau tracé considéré, en conservant les angles de visions précédemment obstruées. Enfin, le processus est répété pour chaque ancien symbole vers le nouveau nœud (ou nœud mis à jour), en conservant les règles définies précédemment pour la création d'un graphe complet. Ce processus permet ainsi de compléter l'étape de reconnaissance à chaque nouveau tracé, permettant de réaliser à chaque étape un nombre minimal d'opération sur ces graphes.

3.2.6 Bilan de la création des hypothèses de graphes de segmentation

À la fin des quatre phases, nous obtenons un ensemble complet de graphes hypothèses représentant l'opération arithmétique saisie. Les nœuds correspondent à des hypothèses de symboles, chacun composé d'un ou de multiples tracés, avec un score de reconnaissance associé. Les liens entre symboles représentent les relations mathématiques, elles-mêmes évaluées suivant la qualité de correspondance avec les paysages flous associés.

Les différentes hypothèses de graphes représentent alors chacune une possible saisie de l'enfant. Nous verrons dans la prochaine section les stratégies d'exploration de ces hypothèses afin d'aboutir à la meilleure interprétation de la saisie de l'élève. La connaissance de la réponse attendue nous permettra de proposer une méthode de sélection d'hypothèses, notamment en utilisant des appariements itératifs les plus vraisemblables pour un ancrage progressif des éléments.

3.3 Module de mise en correspondance : appariement de graphes d'opérations arithmétiques

Dans cette section, nous présentons la mise en correspondance entre des hypothèses de graphes de segmentation avec le graphe de l'opération arithmétique de la solution attendue. Nous présentons nos contributions permettant de réaliser cette mise en correspondance entre graphes, notamment par l'utilisation de fonctions de coût adaptées à la structure de l'opération arithmétique. On détaille aussi les stratégies permettant de faire face à la complexité de ce calcul sur des graphes pouvant atteindre plusieurs dizaines de nœuds.

3.3.1 Mise en correspondance d'hypothèses de graphes

La mise en correspondance de graphes repose sur l'édition de Graphes (GED) présentée précédemment (cf. Section 2.3.2). La méthode se base sur la définition d'une fonction de coût permettant de transformer des éléments d'un graphe (noeuds, liens) en un autre. En trouvant la meilleure correspondance entre les deux graphes, celle qui minimise ces coûts, il nous sera possible de déterminer les différences entre les graphes élèves et le graphe attendu et de comprendre la saisie de l'enfant. Plusieurs hypothèses pourront être considérées et leur traitement sera détaillé par la suite.

Définition 4 (Graphe élève)**.** Graphe obtenu suite au processus complet de segmentation et de reconnaissance dont les noeuds représentent les symboles mathématiques et les liens représentent les relations mathématiques entre symboles.

Définition 5 (Graphe attendu)**.** Graphe généré à partir de la consigne fournie par l'enseignant, dont les noeuds représentent des symboles mathématiques et les liens les relations mathématiques. Ce graphe est construit à partir de l'algorithme de pose d'une opération afin de constituer la réponse attendue à partir de la consigne.

Fonction de coût sur les graphes d'opérations arithmétiques

À partir des paysages flous représentant chaque relation mathématique, il est possible d'évaluer le positionnement relatif par rapport à un positionnement attendu entre symboles. Pour chaque lien $e(i, j)$, représentant un lien entre les noeuds i et j, l'appartenance à chaque paysage flou est conservée des précédentes étapes de reconnaissance. On peut ainsi procéder au calcul du coût d'édition R entre deux liens $e_{(i_1,j_1)}$ et $e_{(i_2,j_2)}$ des graphes comparés avec :

$$R(e_{(i_1,j_1)}, e_{(i_2,j_2)}) = \sum_{d \in D} ||V1_d - V2_d|| * 100 \tag{3.4}$$

où $e_{(i1,j1)}$ et $e_{(i2,j2)}$ sont chacun des liens respectivement des graphes G1 et G2. En utilisant cette fonction pour calculer la différence entre une paire de liens, il est possible de calculer une matrice de coût qui prend en compte à la fois les labels des nœuds et des liens. Cette matrice est de dimension $(n+m)^2$ où n représente le nombre de nœuds dans le graphe G1 et m le nombre de nœuds dans le graphe G2. Cette matrice est construite telle que présenté :

$$\left[\begin{array}{ccc|ccc} c_{1,1} & .. & c_{1,m} & c_{1\to e} & \infty & \infty \\ .. & .. & .. & \infty & .. & \infty \\ c_{n,1} & .. & c_{n,m} & \infty & \infty & c_{n\to e} \\ \hline c_{1\leftarrow e} & \infty & \infty & 0 & 0 & 0 \\ \infty & .. & \infty & 0 & 0 & 0 \\ \infty & \infty & c_{m\leftarrow e} & 0 & 0 & 0 \end{array}\right]$$

où $c_{i,j}$ est une substitution, $c_{i\leftarrow e}$ une suppression et $c_{i\to e}$ une insertion. On calcule le coût d'édition entre deux noeuds v_i et v_j par la Formule 3.5.

$$c_{v_i,v_j} = d(v_i, v_j) + min_{t \in T} \sum_{e_{(i,i')}, e_{(j,j')} \in t} (R(e_{(i,i')}, e_{(j,j')}) + d(v_{i'}, v_{j'})) \tag{3.5}$$

où $d(v_i, v_j)$ est une fonction qui évalue le coût de transformation d'un nœud vers un autre (cf Formule 3.6) et $R(e_{(i,i')}, e_{(j,j')})$ l'édition des labels des liens, où i' et j' sont des noeuds adjacents respectivement à i et j. T est l'ensemble des combinaisons de tous les liens sortants d'un nœud v_i qui peuvent être réalisées avec les liens sortants d'un nœud v_j et t est la liste de paires de liens correspondantes. Le coût d'édition entre deux noeuds correspond à la meilleur combinaisons (moindre coût) de ces liens.

$$d(v_i, v_j) = \begin{cases} 50 & \text{si } \mu_i \in [-,+] \text{ et } \mu_j \notin [-,+] \\ 10 & \text{sinon si } \mu_i != \mu_j \\ 0 & \text{sinon} \end{cases} \tag{3.6}$$

où μ est le label d'un noeud.

3.3.2 Mise en correspondance partielle des hypothèses de graphes

C'est à partir de cette matrice de coût que nous pouvons calculer, au travers de l'algorithme DF-GED présenté précédemment (cf. Section 2.3.2), le coût d'édition d'un graphe élève vers un graphe attendu. Cependant cet algorithme bien qu'optimisé reste très coûteux, et est impossible à appliquer sur des opérations complètes contenant plusieurs dizaines de nœuds. Nous proposons une première stratégie réalisant une mise en correspondance partielle de sous-ensembles spécifiques du graphe, présentée en figure 3.15. En cherchant une correspondance sur un sous-ensemble dont la présence dans le graphe élève est le plus probable (les opérandes à poser de la consigne), et en trouvant cette correspondance, il serait alors plus facile de mettre en lien le reste des symboles de l'opération. De plus, en confrontant chaque hypothèse du graphe élève avec le graphe attendu, il est possible de pré-sélectionner efficacement l'hypothèse ayant la meilleure correspondance partielle pour servir de base à la suite de l'appariement, afin d'éviter de réaliser un appariement coûteux sur de multiples graphes.

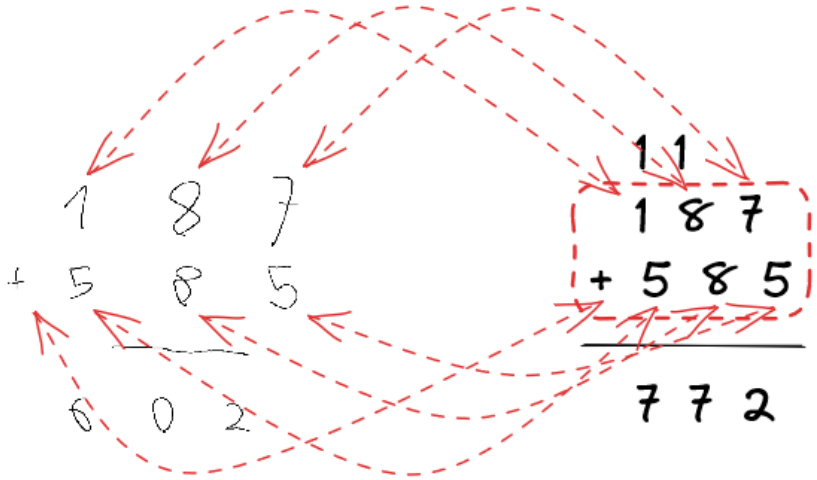

FIGURE 3.15 – Une addition où un premier sous-ensemble de nœuds est extrait pour réaliser la correspondance. Ce sous-ensemble correspond aux nombres composant la consigne de l'opération. En réduisant l'ensemble de recherche, il est possible de trouver efficacement une correspondance pour le sous-graphe au sein du graphe complet.

On se retrouve cependant confronté à deux problématiques. La première est qu'il n'est pas forcément possible de trouver un sous-ensemble de nœuds qui sera efficacement mis en correspondance sur le graphe élève. En effet le graphe élève peut toujours contenir une très grande quantité de nœuds qui doivent être traités. De plus, si une correspondance n'est pas

trouvée sur ce sous-graphe, alors il reste nécessaire de réaliser l'analyse du graphe complet. Nous proposons de résoudre ces problématiques avec une représentation intermédiaire de graphes de lignes pour les deux graphes comparés, et en détaillant un processus itératif permettant d'éviter la génération d'hypothèses superflues pour le processus.

3.3.3 Décomposition des graphes en sous-graphes de nombres (lignes)

Nous proposons donc une décomposition des graphes d'opérations arithmétiques permettant de hiérarchiser la mise en correspondance des graphes. L'objectif est de nous appuyer sur la structure bi-dimensionnelle de l'opération. Une opération arithmétique étant en effet composée de nombres, opérandes et résultats, représentés par des lignes de chiffres, nous introduisons ainsi le concept de "graphes de lignes". En réalisant ainsi une décomposition et une correspondance ligne à ligne, il est possible d'accélérer grandement les traitements proposés.

Décomposition de l'opération

Une opération est constituée de plusieurs lignes : la consigne à recopier (les opérandes), un résultat et de potentielles opérations intermédiaires (multiplication). Tous ces éléments peuvent être plus ou moins bien alignés, mais les nombres seront normalement écrits de la gauche vers la droite, et composés d'éléments considérés comme écrit temporellement proche au sein de l'opération. En utilisant les relations spatiales entre symboles définies précédemment, il est possible de constituer de nouveaux nœuds représentant les lignes de l'opération. Nous allons fusionner plusieurs noeuds en un noeud représentant une ligne de l'opération. La figure 3.16 illustre le processus appliqué sur une addition spécifique. Tout d'abord, un nœud est sélectionné aléatoirement afin de composer le premier ensemble de symboles. On considère que des symboles voisins appartiennent sémantiquement à la même ligne s'ils ont été écrits peu de temps les uns après les autres et avec un alignement gauche-droite. Les symboles proches spatialement avec ces relations mais écrits avec un grand écart temporel sont effectivement principalement des retenues ou des reports réalisés dans le cadre de l'opération. En utilisant les relations spatiales "Gauche" et "Droite" définies grâce aux paysages flous, on relie les symboles adjacents contenant ces relations sans un écart temporel important pour former un nœud composé. Ce processus est répété avec les voisins jusqu'à ce qu'aucun nouveau voisin ne soit découvert (figure 3.16 (b)).

On sélectionne ensuite un nouveau nœud qui n'a pas encore été regroupé afin de répéter l'opération. Quand tous les nœuds ont été traités, alors le graphe de lignes est composé (figure 3.16 (d)). De la même manière, il nous est possible de générer un graphe de lignes à partir du graphe attendu avec ce traitement. Cette représentation contient ainsi un nombre très restreint de nœuds composés, et chaque nœud composé contient une information plus importante sur la structure de l'opération que des symboles isolés. Ces lignes héritent par ailleurs des relations des nœuds enfants provenant du graphe initial.

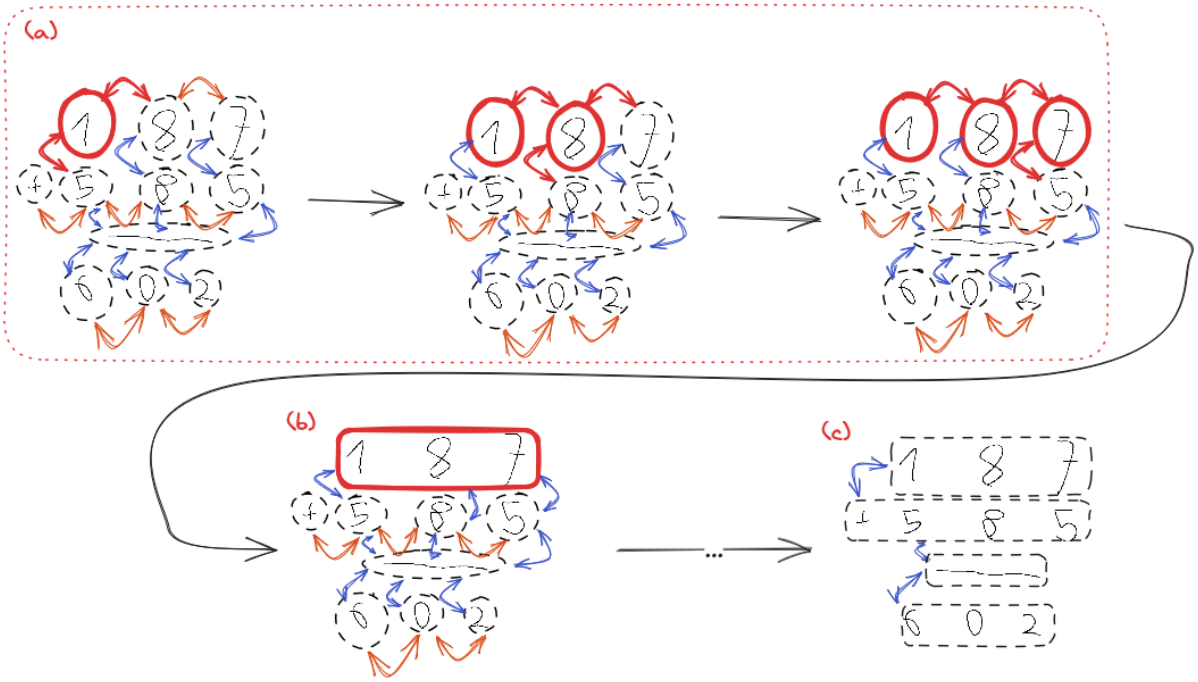

FIGURE 3.16 – Une addition avec initialement 11 nœuds. (a) Un premier nœud aléatoire est sélectionné et ses relations adjacentes sont explorées. Le processus est répété pour chaque voisin Droite/Gauche jusqu'à ce qu'aucun nouveau voisin n'existe pour créer une ligne (b). Le processus est ensuite réalisé jusqu'à ce que tous les nœuds correspondent à une ligne. Le graphe résultat contiens 4 nœuds (c).

1er cycle : mise en correspondance ligne à ligne

À partir de cette nouvelle représentation sous forme de graphes de lignes, il est possible d'appliquer à nouveau l'algorithme DF-GED (cf. Section 3.3.2). Afin de réaliser un calcul de coût propre à cette représentation, chaque sous-graphe est remplacé par une chaîne de caractères en utilisant l'ordre Gauche-Droite. On peut ainsi appliquer une distance de

Levenshtein pour calculer le coût d'édition de deux séquences de symboles. Si deux lignes ont un coût de 0, alors il devient possible de réaliser un appariement nœud composé à nœud composé, et d'ajouter cette information sur le graphe de symboles. Remarquons que la présence de l'opérateur n'étant pas garantie dans la réalisation d'une opération, son coût d'édition est fixé à 0 afin d'éviter d'empêcher la mise en correspondance exacte de deux nœuds composés suite à son absence. Cependant si au moins un des nœuds à l'intérieur de ce sous-graphe ne trouve pas de correspondance, alors les autres symboles ne seront pas appariés à cette étape.

2ème cycle : mise en correspondance symbole à symbole

Après le premier cycle de mise en correspondance ligne à ligne, nous effectuons un deuxième cycle de mise en correspondance sur le graphe de symbole à symbole avec de nouveaux coûts (Equation 3.7). Si une correspondance était trouvée dans le premier cycle, nous fixons les appariements des symboles les composants. Cette stratégie permet d'élaguer une partie importante de l'arbre de recherche. Cela permet à l'algorithme DF-GED d'avoir moins de nœuds à apparier et donc d'accélérer l'appariement.

$$d(v_i, v_j) = \begin{cases} \infty & \text{si trouvé et } f_i ! = f_j \\ 50 & \text{sinon si } \mu_i \in [-,+] \text{ et } \mu_j \notin [-,+] \\ 10 & \text{sinon si } \mu_i ! = \mu_j \\ 0 & \text{sinon} \end{cases} \tag{3.7}$$

où μ sont les labels des noeuds, et f l'idenfiant affixé pour un noeud lorsqu'une correspondance parfaite a été trouvée dans une étape précédente.

3.3.4 3ème cycle : exploration de nouvelles hypothèses de segmentation par backtracking

On le sait, des erreurs de segmentations, de reconnaissance ou des relations peuvent être introduites par les classifieurs. Nous avons proposé une méthode permettant de générer plusieurs hypothèses de graphes enfants. Nous proposons maintenant dans ce 3ème cycle de tirer profit de la mise en correspondance trouvée au cycle 2 pour réaliser des itérations de backtracking afin d'explorer si besoin d'autres hypothèses.

Cette étape de backtracking est présentée dans la figure 3.17. Un exemple où une

mauvaise segmentation sur un symbole au cycle 1 et 2 peut être remise en question et corrigée au cycle 3. L'idée est de tirer profit de la correspondance partielle obtenue en cycle 1, suite à l'appariement ligne à ligne. L'opération contient ainsi un sous-ensemble de symboles qui sont considérés comme corrects vis-à-vis de l'opération attendue. Il est ainsi possible d'explorer de nouvelles hypothèses uniquement sur les sous-ensembles de symboles non trouvés à cette étape. Les tracés qui n'ont pas pu être appariés vont alors être reconsidérés par le classifieur de segmentation, puis de labélisation de symboles et de relations avec un seuil de segmentation τ graduellement plus important qui permettra de générer de multiples hypothèses de reconnaissance de graphes élèves (Cf. Section 3.2). A chaque étape de backtracking, un sous-ensemble d'hypothèses est généré, puis mis en correspondance en itérant sur le cycle 1 et 2. Si une correspondance parfaite est trouvée, ces symboles jusqu'alors non fixés se retrouvent fixés dans l'opération. Il est possible de répéter ces étapes jusqu'à un seuil τ considéré limite.

Lors d'une itération au cycle 3, si plusieurs nouveaux graphes hypothèses sont retournés par le système, alors l'hypothèse ayant le coût de transformation le plus bas est gardée pour l'appariement complet. Les autres hypothèses sont rejetées et ne seront plus considérées. La condition de fin est atteinte si :

— l'intégralité du graphe a trouvé une correspondance ;
— le seuil de segmentation est atteint ;
— aucune nouvelle hypothèse n'est générée.

3.3.5 Bilan de la mise en correspondance de graphes

Dans cette section, nous avons présenté le module de mise en correspondance entre deux graphes représentant des opérations arithmétiques qui s'appuie sur l'algorithme DF-GED. Des fonctions de coût spécifiques aux opérations arithmétiques ont été proposées. Afin de contenir l'évolution de la complexité de mise en correspondance lorsque la taille des graphes devient importante, nous avons proposé une stratégie de décomposition des opérations en sous-graphes de lignes. Une première mise en correspondance (cycle 1) est réalisée sur ces sous-graphes. Cette première mise en correspondance peut ensuite être complétée (cycle 2) avec un appariement symbole à symbole. Enfin, pour traiter les cas les plus complexes où l'appariement n'aurait pas abouti, nous avons mis en place une stratégie de backtracking pour exploser de nouvelles hypothèses de segmentation (cycle 3). On obtient ainsi, comme présenté dans la figure 3.18 la correspondance entre un graphe hypothèse final et le graphe attendu.

Dans la prochaine section nous présenterons comment il est possible d'exploiter cette mise en correspondance pour la génération de premiers feedbacks.

3.4 Module de génération de feedbacks à l'élève

Dans cette section, nous présentons brièvement les principes permettant d'interpréter les résultats de la mise en correspondance entre le graphe élève et le graphe attendu. Cela va se formaliser par la génération d'une liste de feedbacks intelligibles qui pourront par la suite constituer les retours pédagogiques transmis à l'élève. Ces feedbacks sont générés par un ensemble de règles sémantiques simples respectant les règles algorithmiques d'une opération arithmétique. Nous rappelons que c'est ici une première version de feedbacks qui est proposée afin de compléter l'analyse des opérations. Pour être pertinent pour des élèves, un ensemble de travaux sur l'interface resteraient à réaliser, à partir de l'interprétation fournie des erreurs observées.

3.4.1 Règles sémantiques et algorithmiques

Il est essentiel de ne pas faire de retours erronés à l'élève. Si un élève a correctement complété son opération, il est inadéquat de lui soulever de potentielles erreurs qui pourraient être liées à une mauvaise reconnaissance de notre système. Ainsi la présentation ou non des feedbacks suit un ensemble de règles. Si le résultat du calcul proposé par l'élève est reconnu comme correct, alors un feedback de "validation" sera donné à l'élève : par exemple "Bravo, tu as calculé le bon résultat !". En effet, dans ce contexte, même si on constate des différences mineures tel qu'un mauvais alignement avec la solution attendue sur le reste de l'opération, il est plus vraisemblable que cela soit dû à une mauvaise reconnaissance du système. Si le résultat du calcul proposé par l'élève est reconnu comme non correct, alors un balayage des problèmes identifiés est effectué pour en déduire les feedbacks les plus adéquats. Chaque colonne de l'opération est analysée en commençant par les unités les plus faibles :

— Règle n°1 : Lorsqu'une erreur est identifiée sur le résultat d'une colonne, alors un feedback est généré. Le feedback dépendra de la nature des erreurs constatés sur la colonne.

— Règle n°2 : Si un symbole est manquant au sein des opérandes, alors un feedback de mauvais alignement ou un feedback de mauvaise recopie pourra être retourné.

— Règle n°3 : Si un mauvais symbole est reconnu au sein des opérandes, alors un feedback de mauvaise recopie sera retournée.
— Règle n°4 : Si une retenue ou un report était attendu mais absent, alors l'information adéquate sera retournée.
— Règle n°5 : Enfin, si aucun de ces éléments n'intervient dans l'opération, une erreur de calcul sera retournée.

La figure 3.19 présente le balayage effectué pour plusieurs opérations, chacune retournant une valeur différente

3.5 Bilan

Dans ce chapitre, nous avons présenté notre solution d'analyse d'opérations arithmétiques. Le système, décomposé en différents modules, permet de manière itérative la reconnaissance des éléments de l'opération puis leur mise en correspondance afin de générer des feedbacks à l'élève. Ces feedbacks sont une base permettant d'apporter des retours pédagogiques personnalisés aux élèves en fonction des erreurs détectées. Nous avons proposé plusieurs contributions :

— sur la nature des graphes générés mais aussi sur la possibilité de générer de multiples hypothèses afin d'éviter des erreurs de reconnaissance du système. En tirant profit de notre contexte applicatif, nous sommes à même d'utiliser la solution attendue et la correspondance liée afin de remettre en cause les hypothèses de reconnaissance.
— une nouvelle décomposition en sous-graphes de lignes, et en réalisant lors d'une étape intermédiaire une mise en correspondance entre ces sous-graphes, il nous est possible d'aboutir à un appariement complet sur de larges graphes en un temps raisonnable.
— des règles sémantiques appliquées sur le graphe permettent de générer des feedbacks intelligibles permettant d'expliquer les différences observées en les formalisant en erreur de réalisation. C'est sur cette base qu'il sera par la suite possible de réaliser un parcours pédagogique permettant aux élèves la correction de leurs erreurs.

Nous verrons dans le prochain chapitre l'évaluation des différents modules du système, tant sur les scores de reconnaissance au niveau des symboles ou des relations, que des pertinences de la mise en correspondance de graphes de nos différentes stratégies. Nous évaluerons aussi l'impact de ces dernières tant sur la qualité des feedbacks générés que sur la capacité d'apporter ces feedbacks dans un contexte applicatif quasi-immédiat,

en contraignant le temps d'exécution sur nos différentes opérations. Ces éléments nous permettront de conclure quant à la pertinence des contributions mises en avant dans ce chapitre.

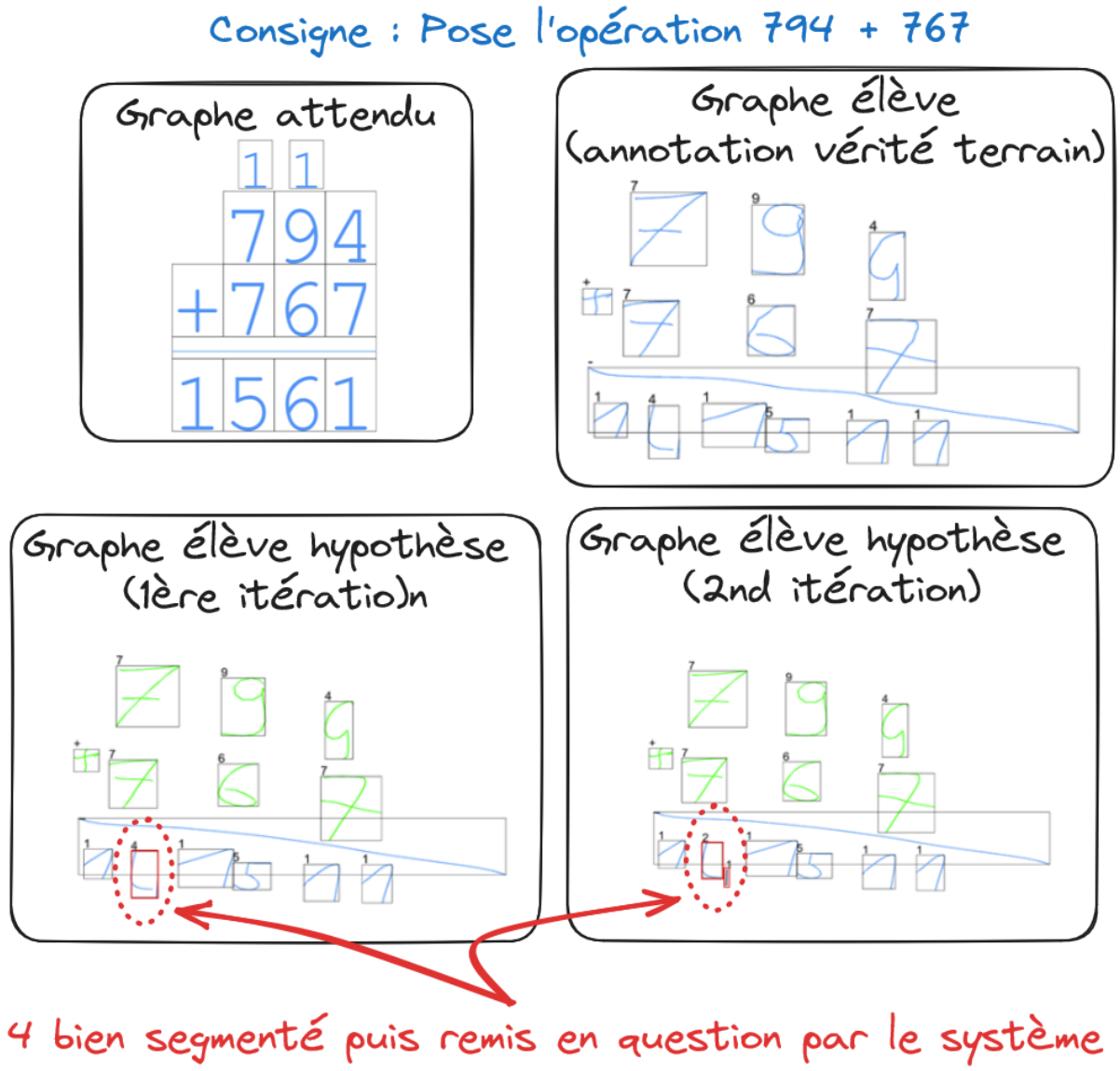

FIGURE 3.17 – Backtracking appliqué sur une addition (a), avec l'annotation de la vérité terrain de la saisie élève (b). Le chiffre 4 est initialement bien segmenté par le système (c), mais n'a pas une mise en correspondance parfaite. Une seconde hypothèse de segmentation est générée (d), produisant une mauvaise segmentation du '4'. Cependant la mise en correspondance de la deuxième hypothèse est considéré meilleure.

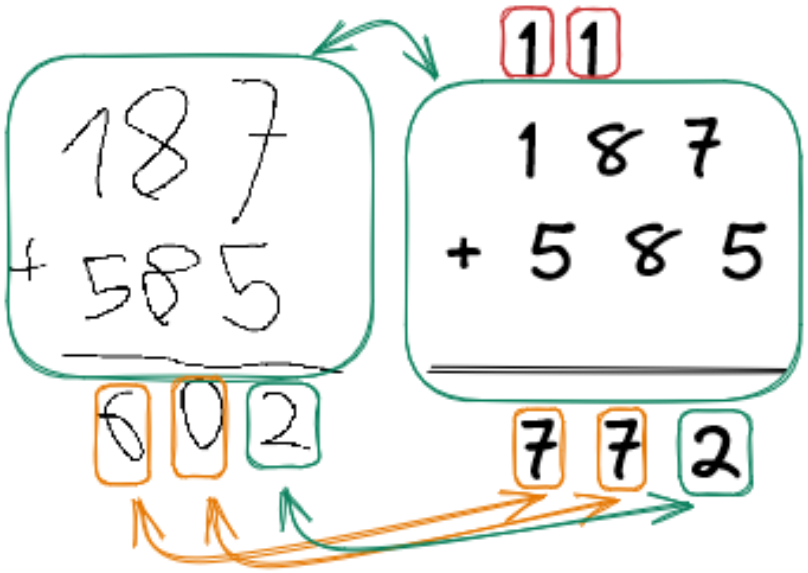

FIGURE 3.18 – Le résultat de correspondance entre un graphe hypothèse et un graphe attendu. Les opérandes de l'opération trouvées avec les bons labels, les chiffres du résultats sont trouvés dont deux avec un label innatendu (noeuds oranges). Deux noeuds, ici les retenues, ne sont pas retrouvés dans le graphe élève (noeuds rouges).

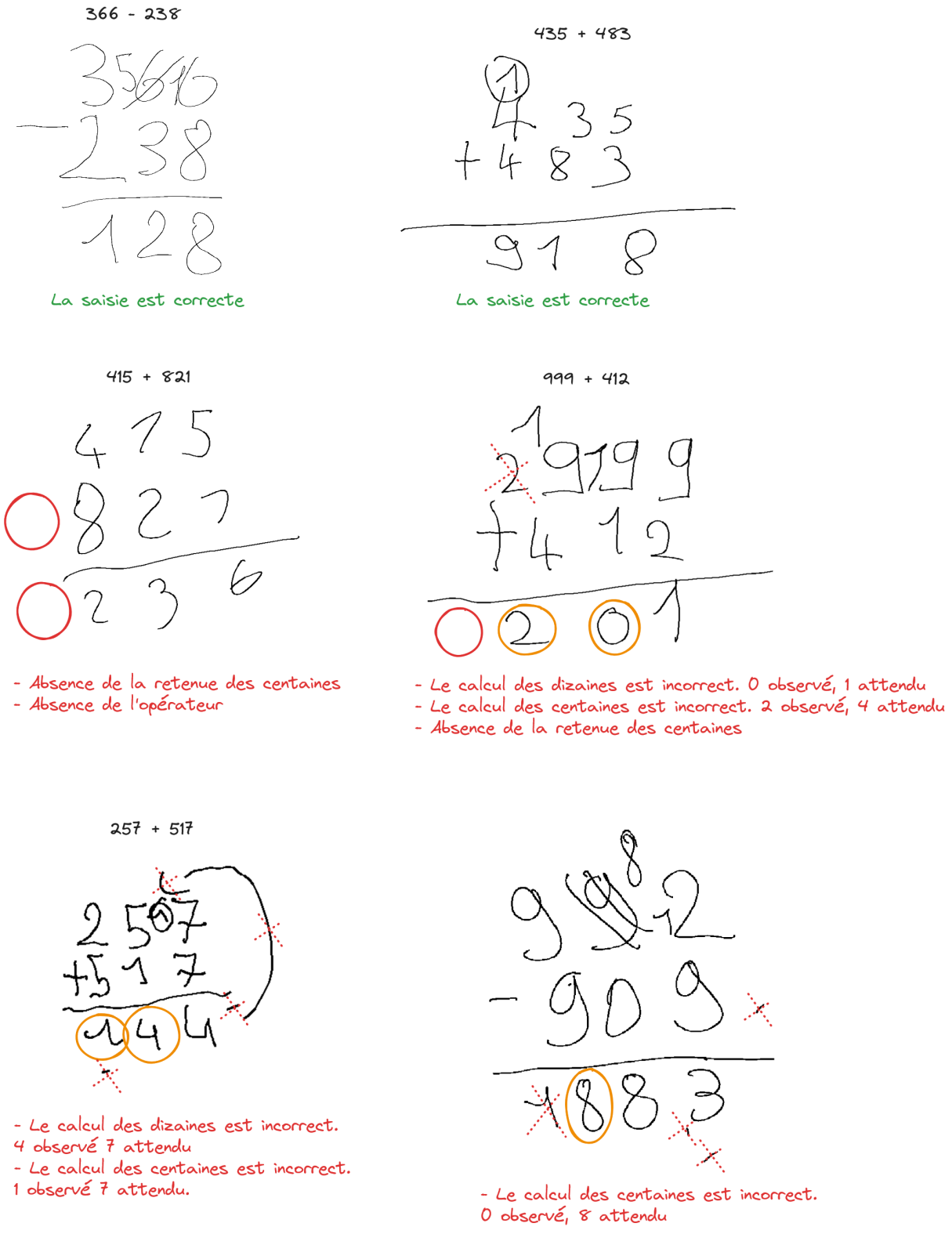

FIGURE 3.19 – Un ensemble d'opérations sur lesquelles les règles sont appliquées pour produire les feedbacks. L'application d'une ou plusieurs règles sont nécessaires afin d'observer différentes typologies d'erreurs. Les symboles barrés sont des tracés innatendus ne trouvant pas de correspondance mais ne produisant pas de feedbacks car pouvant être des tracés de guidage ou des tracés non-voulus par l'élève à cause de mauvais contacts avec l'écran.

Chapitre 4

Expérimentations

Dans ce chapitre, nous introduisons les jeux de données qui ont été utilisés pour l'apprentissage de nos modèles et pour l'évaluation des différentes étapes du système. Les expérimentations réalisées y sont détaillées, en y confrontant quand c'est possible nos résultats à l'état de l'art associé. Nous mettons en exergue nos différentes contributions à la fois sur la conservation d'une bonne reconnaissance de l'ensemble des opérations arithmétiques d'une part, et sur la réalisation de l'analyse avec un temps d'exécution restreint afin de respecter notre problématique principale de la quasi-immédiateté des feedbacks d'autre part.

Sommaire

4.1 Contexte expérimental

Jeux de données d'Opérations Arithmétiques Manuscrites (OAM)

Les différentes données ont été collectées dans différentes classes de primaire, les niveaux s'échelonnant du CP au CM1. Des opérations adéquates avec le niveau attendu par les élèves leur ont été proposées. Ces dernières étaient générées aléatoirement suivant un ensemble de règles (pas de retenues, avec retenues, avec retenues en cascade, dépasse les centaines ou les milliers...) afin d'obtenir une diversité dans les types d'opérations, les réalisations et les erreurs des élèves. En supplément, des opérations ont été saisies par des adultes afin de compléter un ensemble d'apprentissage et de disposer de davantage d'opérations pour réaliser nos expérimentations. Les jeux de données publiques de CROHME 2016 [Mou+16] sont aussi utilisés pour renforcer nos modèles d'apprentissage pour la reconnaissance de symboles. Nous détaillons ensuite la séparation de ces jeux pour chaque module d'apprentissage. Des opérations correctement saisies, avec des tracés inattendus ou avec de multiples typologies d'erreurs peuvent être observées dans les figures 5.1 (correctes), 5.2 (mauvaises additions) et 5.3 (mauvaises soustractions) présentes en annexe.

On retrouve dans la table 4.1 la répartition des données collectées. L'ensemble d'apprentissage a été filtré afin de ne contenir que des opérations saisies correctes et principalement des opérations d'adultes. Les erreurs n'aident pas dans l'apprentissage des différents modèles, et cela nous permet de confronter les performances de notre système en test à un plus grand nombre d'opérations compliquées. En effet lorsqu'une opération est correctement saisie (bonne quantité de symboles, bons symboles), alors le traitement est d'autant plus facilité. Ainsi l'ensemble de test contient le maximum d'opérations avec des erreurs et les différentes observations que l'on peut faire sur ces typologies d'erreurs. La répartition et la fréquence d'apparition par taille (nombre de symboles uniques attendus dans l'opération) dans la base de données sont détaillées dans les expérimentations de la section 4.4.

4.2 Reconnaissance d'expressions mathématiques

Tâche et métrique d'évaluation Dans un premier temps, nous souhaitons évaluer la pertinence des paysages flous pour reconnaître les relations entre symboles mathématiques, et modéliser la structure complète d'une opération. Pour cela, on va s'intéresser

TABLE 4.1 – table présentant le jeu de données Opérations Arithmétiques Manuscrites (OAM), avec le nombre d'opérations et de scripteurs différents, ainsi que la quantité de typologies d'erreurs observées dans les opérations. Seules les erreurs que l'on souhaite identifier par des feedbacks sont identifiées.

	Ensemble d'entraînement	Ensemble de test
Nombre d'opérations arithmétiques	200	200
Nombre de scripteurs	28	24
Nombre de symboles	2908	3285
Nombre d'opérations avec mauvais symbole	-	83
Nombre d'opérations avec symbole manquant	-	122

à la tâche de la reconnaissance d'expressions mathématiques afin d'évaluer les caractéristiques liées aux paysages flous par rapport à l'état de l'art. CROHME 2016 [Mou+16] a introduit une sous-tâche de reconnaissance de structure d'une expression mathématique à partir des **symboles segmentés et labélisés**. L'objectif est de retrouver la structure de l'expression à partir de ces éléments, et on évalue les performances de reconnaissance de la structure avec une marge de 2 erreurs de relation au sein d'une même expression.

Apprentissage d'un modèle de relations inter-symboles. Pour le classifieur de relation, on a choisi d'utiliser des Random Forest en optimisant les méta-paramètres avec une grille de recherche et une cross-validation sur le jeu de données CROHME 2014 [Mou+14]. 90% des données sont utilisées pour l'apprentissage et 10% sont conservées pour la validation du modèle. On fait varier le nombre de forêts entre $20, 30, 40, 50, 100, 200$ et la profondeur maximale d'un arbre entre 30, 40, 50, 100, la meilleure combinaison de ces paramètres étant $n_{tree} = 100, depth = 50$ sur l'ensemble de validation.

On compare plusieurs variantes de ce modèle avec des jeux de caractéristiques différents :

— (a) en utilisant uniquement les caractéristiques des paysages flous (41 caractéristiques floues extraites grâce aux paysages flous (GVF))

— (b) en utilisant en plus la typologie des symboles identifiés par notre classifieur de symboles (41 floues + 16 caractéristiques basées sur la nature des symboles : 57). Cela permet d'évaluer que la sortie d'un simple classifieur de symboles et l'évaluation de la typologie de ces symboles permet d'améliorer les performances du système. Ce classifieur est un SVM appris en utilisant les caractéristiques HBF49 [DA13], permettant d'obtenir un score de classification de 83% sur le jeu de données

CROHME 2014 sur les symboles isolés.

— (c) en utilisant la typologie des symboles de la vérité terrain en entrée (41 floues + 16 symboles : 57)

— (d) en utilisant en supplément à (c) des caractéristiques géométriques pour la classification des relations (41 floues + 16 symboles + 10 géométriques : 67)

Résultats expérimentaux La table 4.2 présente les résultats obtenus pour chaque version du système utilisant des caractéristiques différentes pour la classification des relations. On observe que l'utilisation de la typologie des symboles observés permet effectivement d'améliorer la reconnaissance de la structure de l'opération, puisque cette typologie permet d'éviter d'associer des relations impossibles entre paires de symboles (par exemple une relation exposant avec une parenthèse droite).

TABLE 4.2 – Taux de reconnaissance des relations (structures) d'une expression mathématiques utilisant notre classifieur avec un ensemble de caractéristiques différents sur le jeu de données CROHME 2014 [Mou+14].

	Nombre de caractéristiques	**Taux de reco.** Expression
(a) GVF	41	69.44
(b) + symboles classifiés	57	72.26
(c) + vérité terrain des symboles	57	77.81
(d) + caractéristiques géométriques (GVF+)	67	**78.43**

Positionnement par rapport à l'état de l'art Les résultats présentés dans la table 4.2 utilisent le système (d) pour la sous-tâche de reconnaissance de structure avec les symboles fournis. Le taux de reconnaissance de la structure représente la quantité d'expressions dont la bonne structure est retrouvée par le système : c'est-à-dire que les liens entre symboles permettant la construction exacte de l'expression LaTeX attendue sont trouvés. Le taux de reconnaissance "Structure + Labels" représente le même score, avec la spécificité que la relation mathématique définie entre les symboles doit aussi être exacte. Les colonnes "<= 1 err" et "<= 2 err" représentent la quantité d'expressions correctes où au maximum un ou deux liens sont erronés, afin d'évaluer combien d'expressions contiennent peu d'erreurs de reconnaissance.

MyScript obtient les meilleurs résultats en utilisant un jeu de données privé pour l'apprentissage de ses modèles [Mah+19]. Nous obtenons malgré tout le 3ème meilleur résultat,

au niveau du système WIRIS [Mou+16]. Soulignons que nous permettons dans notre approche des structures d'expressions impossibles car aucune grammaire n'est utilisée pour assurer que l'expression mathématique obtenue est valide. Mais pour ne reconnaître uniquement des expressions valides (challenge CROHME), des approches telles que WIRIS utilisant des grammaires sont bien plus adaptées, puisque les expressions mathématiques du jeu de données sont toutes correctement constituées. Rappelons que pour notre tâche principale qui concerne l'analyse d'opérations arithmétiques d'élèves, nous devons traiter des opérations non valides. Cette expérimentation permet de conclure que le jeu de caractéristiques proposé permet une reconnaissance qualitative de tout type d'expressions mathématiques bi-dimensionnelles. Ces travaux ont amené à une publication à ICDAR 2019 [LAM19a].

TABLE 4.3 – Taux de reconnaissance d'une expression mathématique avec les symboles fournis sur CROHME 2014.

Système	**Structure Rec. Rate**	**Structure + Labels**		
		Rec. Rate	<= 1 err	<= 2 err
MyScript* [Mou+16]	**90.67**	**84.38**	**85.90**	**87.62**
Wiris [Mou+16]	86.61	78.80	80.42	82.75
MST [HZ16b]	76.66	67.44	-	-
BLSTM [Jul+15]	69.27	64.81	67.34	70.69
CYK [LN16]	70.99	61.46	63.89	66.84
Notre système	87.76	78.43	81.54	83.28

*Jeu de données d'entraînement privé utilisé. Les autres méthodes utilisent CROHME 2014.

4.3 Reconnaissance d'opérations arithmétiques

Tâche et métrique d'évaluation. À l'instar des expérimentations sur les jeux de données CROHME, l'objectif ici est d'évaluer les performances de notre module de reconnaissance sur les opérations arithmétiques. Soulignons que les opérations étant écrites par des élèves, elles ne seront pas forcément correctement structurées même lorsque mathématiquement correctes (cf. figure 5.1), ce qui rend la tâche d'analyse plus complexe.

On considère l'évaluation sur l'ensemble d'une opération arithmétique : si tous les symboles sont correctement segmentés et labélisés et si les relations attendues sont pré-

sentes et correctement labelisées, alors l'opération est considérée correctement reconnue. Si, pour l'ensemble des symboles de l'opération, une seule erreur est détectée au sein de l'opération, alors l'opération dans son ensemble est considérée mal reconnue. Notre objectif est d'estimer d'une part la performance du système de reconnaissance, et d'autre part d'évaluer l'impact de notre solution itérative consistant à générer de multiples hypothèses de reconnaissance. On évalue la capacité du système à au moins retrouver l'hypothèse attendue dans cet ensemble, en fonction du seuil de segmentation τ défini (cf. Section 3.2.2). Le score de précision présente la quantité d'opérations correctement segmentées et sélectionnés avec une méthode de correspondance exacte, tandis que le score de rappel représente le nombre d'opérations dont au moins une hypothèse correspond à la vérité attendue.

Apprentissage des modèles évalués. Le classifieur de segmentation (Section 3.2.2) utilise une architecture SVM similaire au classifieur de relation. Pour le classifieur de relation (Section 3.2.4), on utilise le même modèle qu'évalué dans la Section 4.2, à savoir une Random Forest avec le jeu de caractéristiques GVF+, mais avec un apprentissage uniquement sur des relations observées dans le jeu de données OAM. Le classifieur de symboles (décrit en Section 3.2.3) utilise l'architecture VGG16. Chaque modèle est appris à partir du jeu d'apprentissage d'opérations arithmétiques, auquel on ajoute les données de CROHME 2016 pour le moteur de segmentation et de reconnaissance de symboles. 90% des données sont utilisées pour l'apprentissage, et 10% sont conservées pour l'ensemble de validation.

Résultats expérimentaux. Soulignons la complexité d'évaluation dans notre contexte applicatif. En effet, les erreurs de segmentation n'auront pas toutes le même impact sur les erreurs de feedback retournées à l'enfant. Sur la figure 4.1, l'opération contient une barre d'opération qui a été tracée en deux temps. La segmentation correcte est l'attribution de ces deux tracés à un unique symbole (a). Néanmoins une mauvaise segmentation (b) pourra quand même produire un feedback correct à l'enfant : le symbole de la barre d'opération n'entre pas dans l'analyse, il sera considéré 'en trop' par le système lors de l'analyse. Cependant pour évaluer la pertinence de notre module de **reconnaissance**, nous considérons toutes les erreurs de segmentation comme des erreurs sur la reconnaissance de l'opération, quelle que soit leur portée sur le feedback.

La table 4.3 permet de constater l'évolution du score de reconnaissance en fonction d'un seuil de segmentation de plus en plus grand. On constate qu'en augmentant ce seuil, et donc le nombres d'hypothèses que l'on génère, on arrive à retrouver des interprétations

correctes. Cela engendre une génération d'un plus grand nombre d'hypothèses qui devront être traitées, avec en moyenne moins de vingt hypothèses générées. Ainsi sans observer une démultiplication inadéquate des hypothèses, il nous est possible de souvent corriger les erreurs de reconnaissance avec la mise en correspondance qui suivra et qui sera développée en Section 4.4.

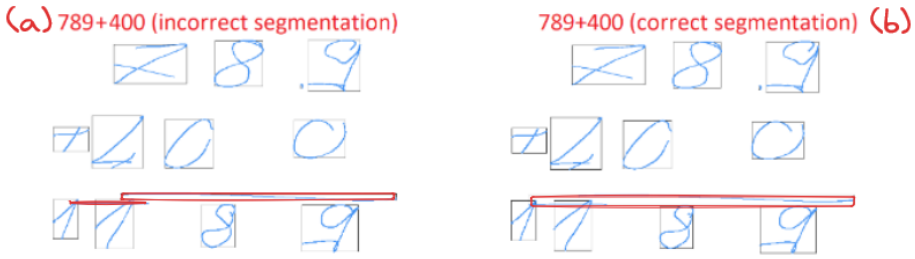

FIGURE 4.1 – Exemples d'une opération avec deux hypothèses de segmentation associées. (a) segmentation non correcte de la barre d'opération. (b) segmentation correcte de la barre d'opération.

TABLE 4.4 – Taux de reconnaissance sur les opérations arithmétiques de la base OAM (table 4.1) test en utilisant l'ensemble du module de reconnaissance proposé (cf Section 3.2), en fonction d'un taux de segmentation évolutif. On voit que pour certaines opérations dont de nombreux tracés se superposent, beaucoup d'hypothèses peuvent être générées.

Seuil de segmentation	Quantité d'hypothèses		Rappel	Précision
	Moyenne	Maximum		
0	1	1	82.5	82.5
0.05	1.16	4	86.0	85.0
0.10	1.7	32	87.0	86.0
0.15	3.04	272	88.5	86.5
0.20	17.42	2880	93.5	90.0

4.4 Mise en correspondance d'opérations arithmétiques

Une fois l'étape de reconnaissance réalisée, le système propose plusieurs graphes représentant des hypothèses de segmentation qui doivent être évaluées par la mise en correspondance avec un graphe attendu. Nous allons nous intéresser à plusieurs apports de

notre système concernant la mise en correspondance d'opérations arithmétiques qui a été décrite en Section 3.3. L'efficacité des fonctions de coût utilisées pour l'appariement des symboles et des relations va influer sur la qualité de la mise en correspondance. L'objectif est d'évaluer la qualité de la mise en correspondance obtenue au terme de l'exécution de chaque approche et d'évaluer la quantité d'opérations dont le traitement aboutit.

1. Mise en correspondance "brute" sur la première hypothèse de segmentation (cf. Section 3.3.1) ;
2. Mise en correspondance "partielle" sur l'ensemble des hypothèses de segmentation (cf. Section 3.3.2) ;
3. Mise en correspondance "ligne à ligne" sur l'ensemble des hypothèses de segmentation (cf. Section 3.3.3).

4.4.1 Stratégie 1 : mise en correspondance "brute"

Tâche et métrique d'évaluation. L'objectif est d'évaluer sur notre jeu de données d'opérations arithmétiques les performances des mises en correspondance suivant nos différentes sous-représentations. Pour toutes ces expérimentations, un temps d'exécution maximum de 5 secondes est observé afin d'estimer la quantité d'opérations que nous ne pouvons traiter en temps raisonnable avec le système. Si une opération atteint ce temps maximum, alors la dernière meilleure correspondance trouvée est conservée pour évaluer le score. L'ensemble des expérimentations est réalisé sur un ordinateur avec un processeur Intel i5-8250U et 8GB de RAM. Il est à noté qu'un sous ensemble de ces expérimentations a été réalisé sur une tablette android afin d'assurer la portabilité du système. Les temps d'exécution observés sur tablette sont du même ordre de grandeur que ceux complets réalisés sur ordinateur. On observe à la fois les scores de correspondance et la quantité d'opérations qui atteignent le temps limite d'exécution pour chaque taille de graphe observée. Le score est calculé sur le nombre de différences observées entre l'opération et la solution attendue par rapport à la vérité terrain. Si une différence est observée, le résultat du système est considéré comme erroné.

Pour cette première tâche, *l'algorithme DFS est appliqué uniquement sur le premier graphe hypothèse généré (cf. Section 3.3.1)*. On compare ainsi la totalité des correspondances paire à paire. On évalue le nombre d'opérations qui ont une correspondance complète et on évalue également le pourcentage de mauvaises correspondances nœud à nœud.

Résultats expérimentaux quantitatifs. Les scores de mise en correspondance en

fonction des tailles des graphes attendus sont présentés sur la figure 4.2 (a). On remarque que pour de petites opérations, on obtient une mise en correspondance correcte la plupart du temps. Cependant avec de plus grandes opérations, la mise en correspondance est souvent incorrecte. On observe sur la figure 4.2 (b) que la quantité d'opérations traitées par cette première version du système est faible dès lors que l'on dépasse des graphes à plus de 10 nœuds. Cette stratégie simpliste permet le traitement en moins de 5 secondes de 47 opérations sur 200. La plupart des opérations qui dont le traitement ne peut aboutir résulte en une mauvaise mise en correspondance retournée par le système, puisque le parcours de l'arbre de recherche n'est pas complété.

4.4.2 Stratégie 2 : mise en correspondance "partielle"

Tâche et métrique d'évaluation. Nous évaluons la même tâche avec la même métrique que la stratégie 1, cette fois-ci pour évaluer la pertinence de *la stratégie de mise en correspondance partielle de graphes, dont l'objectif est d'obtenir plus rapidement une correspondance adéquate*. Une mise en correspondance partielle est appliquée cette fois sur l'ensemble des sous-graphes afin de faire correspondre des parties de l'opération avant d'appliquer l'algorithme DFS sur le graphe complet de symboles.

Résultats expérimentaux quantitatifs. On remarque que le score sur la figure 4.2 (a) est amélioré en comparaison de la stratégie "brute" sans optimisation de mise en correspondance. Cette observation est corrélée avec les résultats présentés sur la figure 4.2 (b). D'avantage d'opérations sont résolues dans le temps fixé, nous passons de 47 à 134 opérations (sur 200) dont le traitement aboutit. La stratégie de mise en correspondance partielle permet d'améliorer les scores sur des graphes sur lesquels le traitement ne pouvait aboutir.

4.4.3 Stratégie 3 : mise en correspondance "ligne à ligne"

Tâche et métrique d'évaluation. Nous évaluons la même tâche avec la même métrique que les stratégies 1 et 2, cette fois-ci pour évaluer la pertinence de la *mise en correspondance à partir de graphes de lignes*. On évalue dans un premier temps (Stratégie 3a, Section 3.3.3) l'algorithme DFS appliqué itérativement sur les hypothèses de graphes de lignes puis sur le premier graphe de symboles avec les correspondances trouvées déjà fixées. On évalue dans un second temps (Stratégie 3b, Section 3.3.4) l'algorithme DFS appliqué dans le processus complet de backtraking sur de multiples hypothèses et sur le

dernier graphe conservé pour l'analyse. Pour la méthode de backtracking, on considère un incrément du seuil de segmentation τ de 0.05% en réalisant un maximum de 5 itérations pour obtenir un $\tau = 0.20$ tel qu'évalué précédemment.

Résultats expérimentaux quantitatifs. On constate ainsi dans les figures 4.2 (a) et 4.2 (b) de meilleurs scores de mise en correspondance, à nouveau corrélés avec un nombre plus importants d'opérations dont le traitement aboutit dans le temps imparti. On remarque ainsi que la segmentation en graphes de lignes permet d'améliorer significativement le nombre d'opérations correctement analysées dans un temps limité (< 5 secondes). Quant à l'optimisation de la stratégie 3b grâce au backtracking sur les hypothèses, on remarque que l'on est capable de retrouver les hypothèses de reconnaissance pertinentes sans sacrifier de temps de traitement grâce à la correspondance approximative de lignes. En traitant l'hypothèse de reconnaissance correcte, le système peut ainsi aboutir à la mise en correspondance attendue. Ainsi notre meilleur système (Stratégie 3b) peut aboutir au résultat attendu de 162 opérations (sur 200) en moins de 5 secondes.

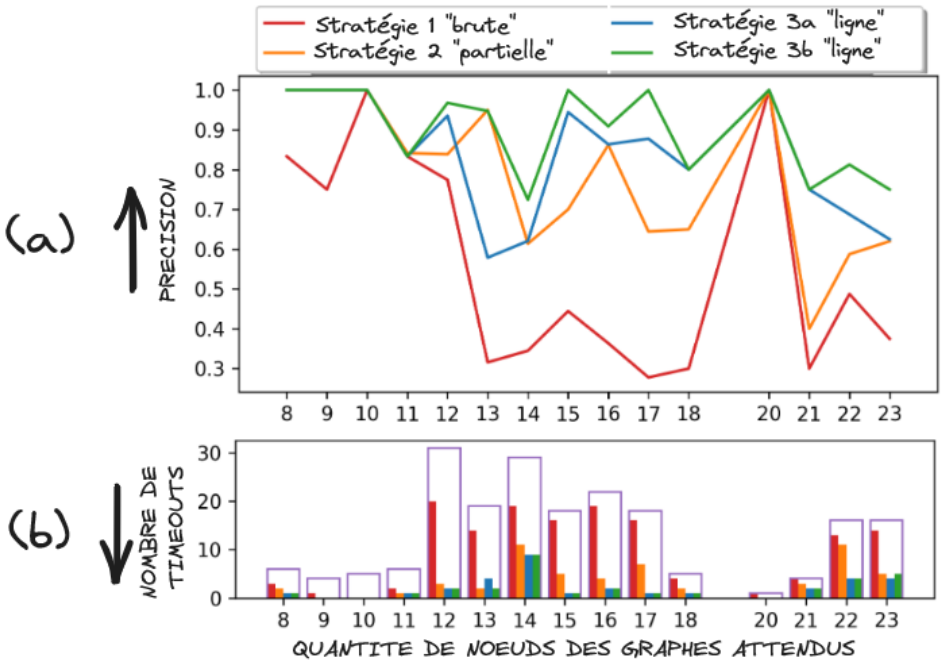

FIGURE 4.2 – Score de correspondance (a) et quantité d'opérations atteignant le temps limite d'exécution (b) pour chaque taille de graphe. La correspondance est considérée correcte si l'ensemble des différences attendues sont trouvées par le système. La quantité maximale du nombre d'opération pour chaque taille de graphe est représentée en violet. Comparaison des Stratégies 1, 2, 3a et 3b.

Résultats expérimentaux qualitatifs. Nous illustrons quelques résultats de façon plus qualitative. Pour la première opération (figure 4.3) le processus itératif permet de trouver une meilleure hypothèse de reconnaissance alors que pour la seconde opération (figure 4.4) le processus itératif amène à la conservation d'une mauvaise hypothèse de reconnaissance. Sur les figures, on présente la consigne (a) qui permet de produire le graphe attendu (b), la vérité terrain de la saisie de l'élève (c), le résultat de reconnaissance de la première itération (d) ainsi que la nouvelle hypothèse générée par le processus itératif (e). Sur la première opération, le système corrige la mauvaise segmentation après une seule itération en trouvant la bonne correspondance. Cependant ce processus sur la seconde opération échoue, car les hypothèses de segmentation n'ont aucune correspondance adéquate avec la solution attendue. La mauvaise hypothèse de reconnaissance a un coût moins élevé de mise en correspondance avec la solution attendue, et se retrouve conservée. Bien que peu représentatif, cet exemple met en lumière le défaut de vouloir absolument s'approcher au mieux de la solution attendue, quitte à prendre des hypothèses de reconnaissance qui détériorent les scores des classifieurs. Une des solutions à apporter pour corriger ces cas pourrait être d'introduire une corrélation entre la détérioration des scores de reconnaissance et l'amélioration des correspondances observées, afin de ne pas chercher des hypothèses si elle n'améliore que de peu la correspondance. Ces travaux ont amené à la publication [LAM21].

4.5 Génération de feedbacks à l'élève

Tâche et métrique d'évaluation. Nous nous intéressons dans cette section à la pertinence des retours qui peuvent être faits suite à la mise en correspondance de graphes. Pour chaque opération, on évalue la liste des feedbacks qui peuvent être retournés en analysant la correspondance des symboles et des relations avec la hiérarchie définie dans la Section 3.4. On vérifie si le système ne renvoie pas des feedbacks non attendus, et si la nature des feedbacks identifie correctement l'erreur commise. On note 5 niveaux de feedbacks pouvant se cumuler pour chaque opération. Pour chaque typologie de feedback, on évalue si notre système les a correctement retournés :

1. Pas d'erreur ;
2. Une erreur sur la saisie d'un chiffre de la consigne ;
3. Une erreur sur l'alignement d'un chiffre ;

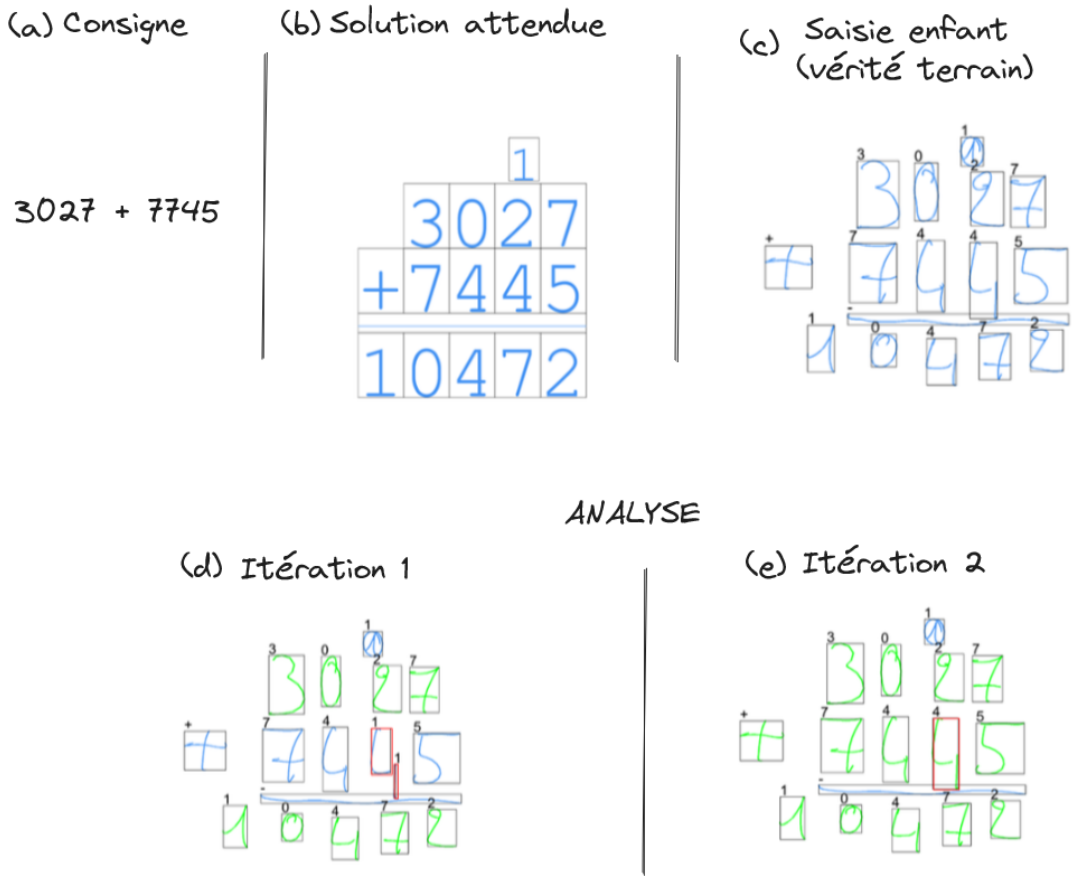

FIGURE 4.3 – Une opération avec une segmentation initiale incorrecte, qui est par la suite corrigée grâce au backtracking permettant de récupérer la bonne hypothèse de segmentation (Stratégie 3b ligne).

4. Une erreur sur le résultat : mauvais résultat ou résultat manquant ;
5. Une erreur sur les retenues ou reports : mauvaise retenue ou report ou absence de la retenue ou du report.

Résultats expérimentaux. Dans le cas d'une opération avec de multiples erreurs d'une même typologie (par exemple plusieurs symboles erronés sur le résultat), les retours du système sont considérés valides uniquement si toutes les erreurs sont retournées. Il ne s'agit pas de simplement trouver une des erreurs du résultat. Ainsi sur les 200 opérations de notre jeu de données, 180 produiront les feedbacks adéquats pour l'élève. La différence entre les 162 opérations dont la correspondance aboutie avec assurance, et les 180 opérations que l'on annote correctement sont dues aux correspondances complètes qui n'aboutiront pas, mais dont la mise en correspondance intermédiaire lorsque le temps

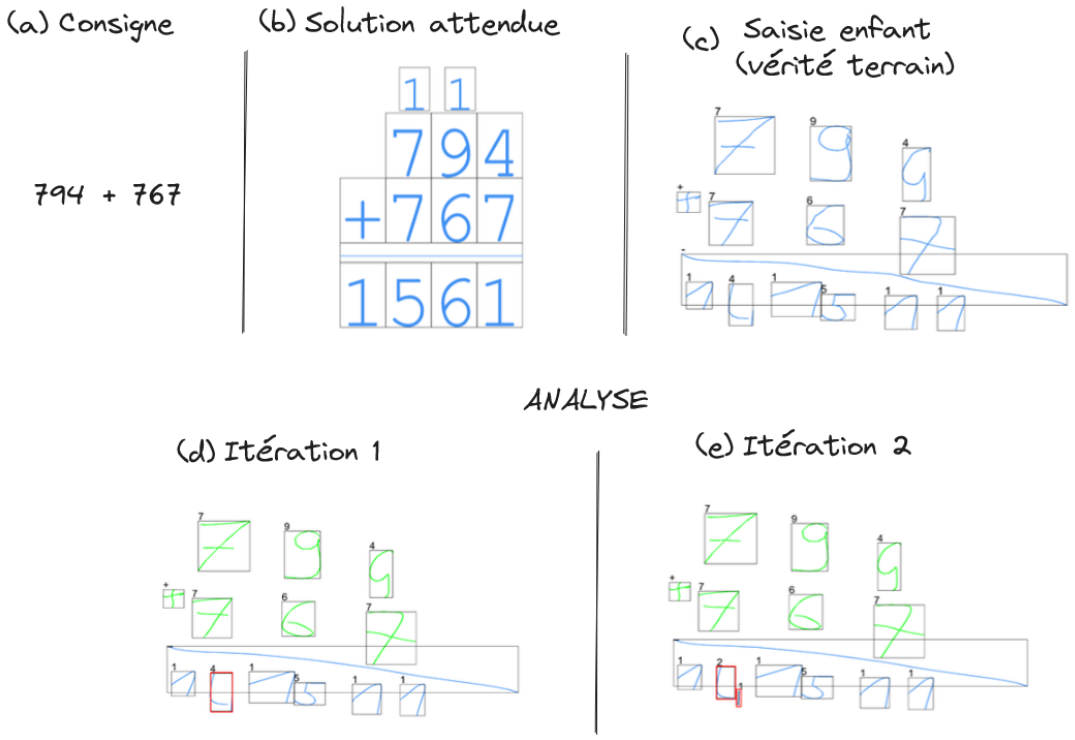

FIGURE 4.4 – Une opération avec une segmentation incorrecte sur le 4 sur la seconde itération qui résulte en une correspondance à un coût élevé mais moindre par rapport à la bonne segmentation attendue, résultant en une mauvaise hypothèse conservée.

limite d'exécution est atteint se trouve être une mise en correspondance permettant d'obtenir le bon feedback. La figure 4.5 montre deux exemples d'opérations, avec la mise en correspondance obtenue pour chaque opération et les feedbacks qui en résultent. Pour l'opération (b) le symbole du résultat est mal reconnu par le système, et donc un feedback non attendu est produit. Les erreurs produites par le système sont en adéquation avec les erreurs de reconnaissance évaluées plus tôt. Dans le cadre où l'opération est correctement reconnue et que la mise en correspondance aboutie, alors le système se comporte comme attendu, comme pour l'opération (a).

Tous ces éléments ont amené à la réalisation d'un premier prototype de logiciel sur tablette, illustré en figure 4.6, où l'élève peut saisir sur une page non restreinte son opération. Nos différentes contributions à la fois sur la construction itérative des hypothèses et sur la recherche par mise en correspondances partielle sur les graphes de lignes permettent la réalisation et l'obtention de ces résultats en quelques secondes après la fin de saisie de l'exercice.

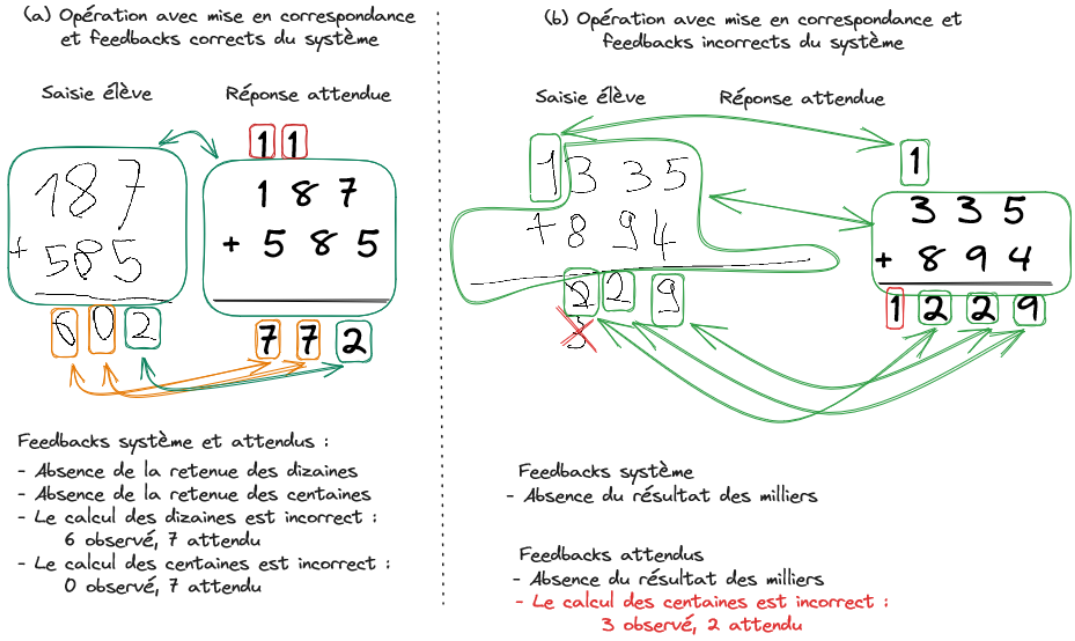

FIGURE 4.5 – Deux opérations arithmétiques avec la mise en correspondance retournée par le système et les feedbacks résultants. Sur l'opération (a), la mise en correspondance est correcte et produit donc un feedback adéquat. Sur l'opération (b) le résultat de l'unité est mal reconnu par le système, qui retourne donc une erreur sur ce même résultat.

4.6 Bilan

Nous avons commencé par évaluer notre système sur un ensemble d'expressions mathématiques pour mesurer la pertinence de nos paysages flous par rapport à des méthodes de l'état de l'art pour la reconnaissance de la structure des expressions. Nous obtenons des résultats au niveau de l'état de l'art pour cette reconnaissance, alors que notre système ne s'appuie pas sur des connaissances sémantiques. Les paysages flous appris à partir des relations mathématiques sont donc une bonne représentation structurelle d'une expression.

En ce qui concerne notre objectif, à savoir analyser des opérations arithmétiques, nous avons pu évaluer notre système sur un ensemble conséquent et varié d'opérations allant de la simple addition observée en CP à des additions de 2 à 3 nombres atteignant les milliers adaptés pour du niveau CM1, en passant par des soustractions saisies suivant les deux modalités existantes, et quelques multiplications. On constate la pertinence de nos différentes stratégies de mise en correspondance. La génération de multiples hypothèses de reconnaissance nous permet dans une grande majorité des cas d'obtenir la bonne re-

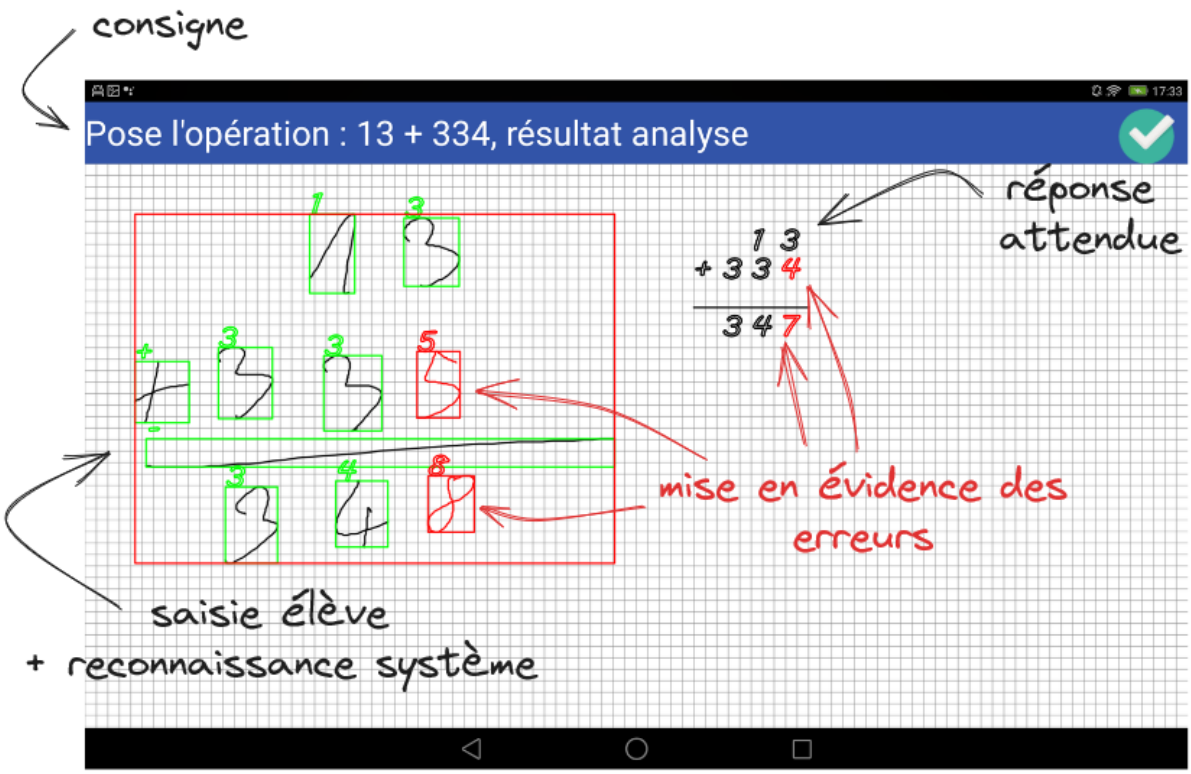

FIGURE 4.6 – Un premier prototype de l'application Kaligo Maths. Ici la conclusion du système sur l'opération écrite dans un environnement non restreint est représentée par la mise en évidence des symboles différents inattendus observés, quasi immédiatement après saisie de l'opération.

connaissance alors qu'elle n'était pas obtenue lors de l'itération initiale. La décomposition en lignes (Stratégie 3) grâce aux caractéristiques visuelles extraites permet une décomposition adéquate des opérations qui ne dénature pas la structure de ces dernières. D'une part, on note une bonne mise en correspondance et une accélération notable de l'appariement complet des graphes par cette stratégie de mise en correspondance par ligne. D'autre part, l'exploration de nouvelles hypothèses de reconnaissance dans les cas de mise en correspondance incomplète permet de corriger des erreurs produites sur la première itération.

Enfin, les feedbacks retournés à l'élève sont pertinents pour 180 sur 200 opérations. Nous pouvons donc conclure en premier lieu que la génération de multiples hypothèses de reconnaissance permet la correction erreurs induites par le système. La représentation en graphes de lignes permet une mise en correspondance performante et accélérée réduit-

sant l'espace de recherche des hypothèses. L'insertion de symboles non-attendus au sein d'une opération, notamment par l'encerclement de retenues ou la séparation des unités de l'opération, n'impacte pas non plus la reconnaissance et la mise en correspondance, ces éléments étant bien rejetés lors de ces différentes étapes. On constate cependant encore quelques difficultés lorsque de multiples erreurs simultanées sont faites par l'élève à plusieurs niveaux de l'opération. Dans ce contexte, trop peu d'éléments peuvent être fixés en amont, et le système n'aboutit pas à une conclusion adéquate dans un temps raisonnable. Ces exemples pourraient être corrigés en assouplissant les conditions de mise en correspondance entre lignes, il serait cependant important de pouvoir toujours remettre en question dans les premières itérations la reconnaissance du système afin d'éviter de valider trop tôt une mauvaise hypothèse de reconnaissance.

Chapitre 5

Conclusion

Sommaire

5.1 Contributions

Les travaux de thèse présentés dans ce manuscrit ont porté sur la conception d'un système permettant l'interprétation d'opérations arithmétiques manuscrites posées à la volée. L'objectif était d'offrir aux élèves un outil d'aide à l'apprentissage des opérations arithmétiques. Pour cela, nous avons travaillé à concevoir un système d'analyse automatisé des opérations qui sera au coeur des logiciels d'e-education développés ensuite par la société Learn&Go associée à cette thèse. Le système devait répondre à plusieurs besoins dont celui de simuler l'approche traditionnelle papier crayon en s'appuyant sur des tablettes numériques accompagnées de stylets afin de permettre la saisie dans un environnement non restreint. L'objectif étant de maximiser les capacités de transfert des compétences de l'élève entre la manipulation de l'outil sur tablette et dans son environnement de travail conventionnel (papier et crayon). En se démarquant des outils existants ne reposant pas sur une saisie manuscrite libre, nous obtenons une meilleure transférabilité entre le support numérique et le support papier. Le système se devait aussi de pouvoir effectuer des retours à l'élève (feedbacks) en un temps quasi-immédiat, afin de garder son attention, de lui apporter de l'aide quand il en a besoin et éviter la reproduction récurrente d'une même erreur. Enfin, il était nécessaire de pouvoir apporter une analyse pertinente en permettant la création de feedbacks personnalisés en fonction de la nature de l'erreur commise, afin de le guider dans sa correction. Pour cela il fallait pouvoir comprendre et expliquer les différences constatées entre la réalisation de l'élève et la réalisation attendue correspondant à la solution.

Le système que nous avons conçu pour répondre à ces différentes problématiques apporte plusieurs contributions dans le domaine de la reconnaissance d'opérations mathématiques et sur l'utilisation et le traitement de mise en correspondance de graphes pour l'appariement exact de ces derniers. Cela nous permet d'obtenir les retours du système dans une temporalité contenue compatible avec le processus d'apprentissage d'un élève lors de la résolution d'exercices.

Nous avons tout d'abord proposé une représentation adaptée sous forme de graphes de visibilité des opérations arithmétiques. L'utilisation de ces caractéristiques visuelles permet de qualifier les relations entre symboles de l'opération, afin d'évaluer et d'interpréter par la suite de potentielles différences observées. La construction de ce graphe est aussi accompagnée de multiples améliorations permettant de résoudre des erreurs de reconnaissance du système dans un temps contenu. Chaque étape dans le processus de

reconnaissance, la segmentation, la reconnaissance des symboles et des relations permet la création de plusieurs hypothèses de reconnaissance, chacune pouvant être considérée comme une hypothèse de la réalisation de l'élève.

Cela permet d'explorer plusieurs hypothèses vraisemblables pour déterminer ensuite celle qui permettra d'aboutir à la meilleure mise en correspondance. De plus, le processus de création itératif étend les hypothèses générées à partir du graphe de visibilité et engendre un ensemble de graphes qui s'accroit au cours de la saisie. Cela permet d'analyser l'opération en cours de saisie et d'éviter un traitement complet de l'opération a posteriori.

Nous avons ensuite travaillé sur la mise en correspondance de graphes, principalement utilisée jusqu'à présent dans des travaux de classification, afin de réaliser un appariement entre la solution attendue et la réalisation de l'élève. Les contributions à ce niveau sont multiples. L'algorithme réalisant une mise en correspondance exacte est trop complexe pour aboutir à un résultat dès lors que les opérations dépassaient une quantité trop importante de noeuds. Ainsi, nous avons d'abord proposé une évolution de la représentation sous forme de graphes d'opérations arithmétiques grâce aux caractéristiques visuelles des paysages flous, afin de constituer des graphes de lignes, puis adapté des opérations de coûts permettant de réaliser l'appariement sur cette typologie de graphes. Nous avons aussi proposé une stratégie de mise en correspondance partielle sur ces mêmes graphes de lignes afin d'isoler l'hypothèse de reconnaissance la plus vraisemblable. Cela permet de contenir la complexité de l'algorithme sur un ensemble de sous-graphes de taille moindre, et de réaliser l'appariement en un temps restreint.

Afin de corriger les erreurs de reconnaissance, nous avons proposé une stratégie de backtracking qui profite des capacités de mise en correspondance de notre système sur les hypothèses générées pendant la reconnaissance. Initialement, l'ensemble des hypothèses générées est restreint afin de limiter la mise en correspondance sur un trop grand ensemble de graphes. Si les premières mises en correspondances ne sont pas suffisament concordantes, nous cherchons par backtracking de potentielles nouvelles hypothèses de reconnaissance qui pourraient mieux convenir. La réalisation de ce processus itératif de backtraking permet de générer à la demande un sous-ensemble d'hypothèses croissant pour rapidement obtenir un meilleur appariement. Ce processus permet ainsi dans le cas d'opérations denses et complexes, notamment lors de la réalisation de multiples reports pour une soustraction, de réaliser une recherche très locale de nouveaux symboles et de corriger les erreurs de segmentation qui peuvent facilement être observées dans ce contexte. Ce processus itératif d'analyse permet de corriger des erreurs de reconnaissance de façon

efficace pour aboutir à la bonne mise en correspondance dans un temps contraint.

Enfin, nous avons proposé un ensemble de règles simples basées sur le résultat de mise en correspondance de noeuds trouvés afin de générer les feedbacks pour l'élève. Ces règles représentent la construction d'une opération arithmétique, et l'algorithme que doit suivre un élève pour la résoudre. On va ainsi échelonner les différentes typologies d'erreurs dans leur ordre d'impact sur l'opération, et établir une conjonction sémantique entre ces éléments en réalisant une passe verticale sur l'opération. Ainsi l'analyse d'une vraisemblable erreur de calcul sur les centaines sera dépendante de l'observation d'un bon alignement des éléments de la consigne, puis d'une bonne recopie de ces derniers, puis de la présence ou non de retenues ou de report attendus avant de conclure à une simple erreur de calcul.

Toutes ces contributions permettent d'aboutir à des feedbacks pertinents sur les opérations proposées, comme en attestent les résultats de nos expérimentations sur un grand ensemble d'opérations saisies par des élèves de CP à CM1. Ces résultats ont été obtenus sur des opérations de taille très variables pouvant atteindre un nombre important de noeuds. On est capable d'obtenir une analyse pertinente et quasi-immédiate sur la plupart des exemples (180 sur 200 opérations). Le système conçu pendant cette thèse va constituer le moteur d'analyse des opérations arithmétiques manuscrites qui sera intégré dans le développement de la future solution "Kaligo Maths".

5.2 Perspectives

Nous l'avions mentionné dès la présentation du contexte de nos travaux, l'introduction d'un tel outil dans un environnement pédagogique ne peut pas s'arrêter au simple traitement et affichage des erreurs de l'élève. Nous avons respecté le besoin de réaliser ces opérations dans un environnement non restreint, cependant l'objectif pédagogique de telles applications est de pouvoir décharger l'enseignant en identifiant et en expliquant aux élèves leurs erreurs afin de les accompagner dans la correction de ces dernières. On se situe très tôt dans la boucle d'apprentissage, puisqu'il est question des prémisses de l'apprentissage des mathématiques dans des classes d'élèves très jeunes. Des erreurs observées vont souvent être dues à une mauvaise compréhension à l'instant donné des règles qui devaient être appliquées pour l'opération, nous ne pouvons donc pas nous contenter de retourner une liste des erreurs constatées. Ainsi, la première perspective d'évolution de ce système concerne les feedbacks produits. Au fur et à mesure de la progression de l'élève, à la fois en mathématiques mais aussi au sein de son parcours scolaire (CP au CM2), la typologie des feedbacks qui lui seront retournés devront évoluer pour devenir un réel atout pédagogique. D'un côté, dans la manière de les présenter à l'élève : un élève de CP n'a pas la même aisance de lecture qu'un élève de CM2 et ces feedbacks devront donc être compréhensible en fonction de l'élève. De l'autre côté, un élève ayant fait une erreur unique, ou un élève ayant reproduit la même erreur sur plusieurs opérations auront besoin d'une approche dans l'explication différente. Cette recherche devra se faire avec des experts académiques afin de construire la solution la plus pertinente pédagogiquement. Une fois un tel système en place, il serait intéressant de pouvoir évaluer la pertinence pédagogique et la fiabilité du système en établissant des expérimentations auprès d'élèves à différentes étapes d'apprentissage.

Ensuite, nous nous sommes concentrés lors de ces travaux sur des additions, soustractions et multiplications posées. Nous avons constaté que malgré une augmentation significative de la taille des opérations, nous sommes en capacité d'apporter un traitement quasi-immédiat pour obtenir les feedbacks pour l'élève. Cependant, les divisions n'ont pas été traitées en partie car elles représentent des contraintes supplémentaires pour l'interface de saisie, ces dernières pouvant s'allonger considérablement verticalement. De plus, celles-ci peuvent atteindre très rapidement un nombre très important de symboles. Sur des divisions de milliers avec virgules, on peut facilement atteindre les 50 symboles mathématiques, et contrairement aux précédentes opérations observées, la multiplication de

petites sous-opérations rend la décomposition en graphes de lignes moins pertinentes, tant ces dernières restent trop nombreuses. Il serait nécessaire de proposer une nouvelle sous-décomposition temporaire sous forme d'un sous-ensemble d'opérations afin de contenir l'augmentation trop importante de noeuds. À l'instar de notre décomposition en lignes, il serait possible d'appliquer une correspondance adéquate performante grâce aux différentes caractéristiques visuelles afin d'aboutir à une correspondance intermédiaire pertinente, permettant ainsi l'application de ce système sur ces grandes opérations.

Enfin, la pose d'opérations arithmétiques ne représente qu'un sous-ensemble de l'apprentissage des mathématiques lors des Cycles 2 et 3. Outre le domaine de la géométrie, qui nécessite un traitement à part entière, on peut imaginer l'extension de nos travaux pour l'analyse du calcul en ligne, qui est tout aussi important pour les élèves car il permet entre autre de traiter de la décomposition des nombres afin de réaliser des opérations en plusieurs étapes. Cependant, contrairement aux opérations posées, cette résolution de calcul n'est pas restreinte. L'élève peut donc décomposer et écrire son opération de multiples manières, rendant complexe la réalisation d'un appariement comme nous avons pu le faire pour les opérations posées. On ne peut simplement pas analyser le résultat pour le valider ou l'invalider, car l'intérêt du système est de pouvoir expliquer où se situent les erreurs et leur nature ; on devrait ainsi pouvoir identifier une mauvaise décomposition ou l'oubli d'une unité. Une approche pourrait reposer sur la décomposition de l'opération en lignes, afin de restreindre le traitement de l'analyse ligne par ligne. Une équivalence mathématique entre chaque ligne pourrait être étudiée pour identifier les erreurs à chaque étape de l'opération. Il serait alors possible de réaliser un appariement avec un sous-ensemble de décompositions équivalentes pour identifier les potentiels oublis, erreurs de calcul ou mauvaises décompositions. La restriction de cette espace de recherche, à l'instar de la génération de nos hypothèses, permettrait de contrôler l'espace des hypothèses générées considérées, afin de toujours réaliser ce processus dans un temps quasi-immédiat.

ANNEXES : EXEMPLES D'OPÉRATIONS

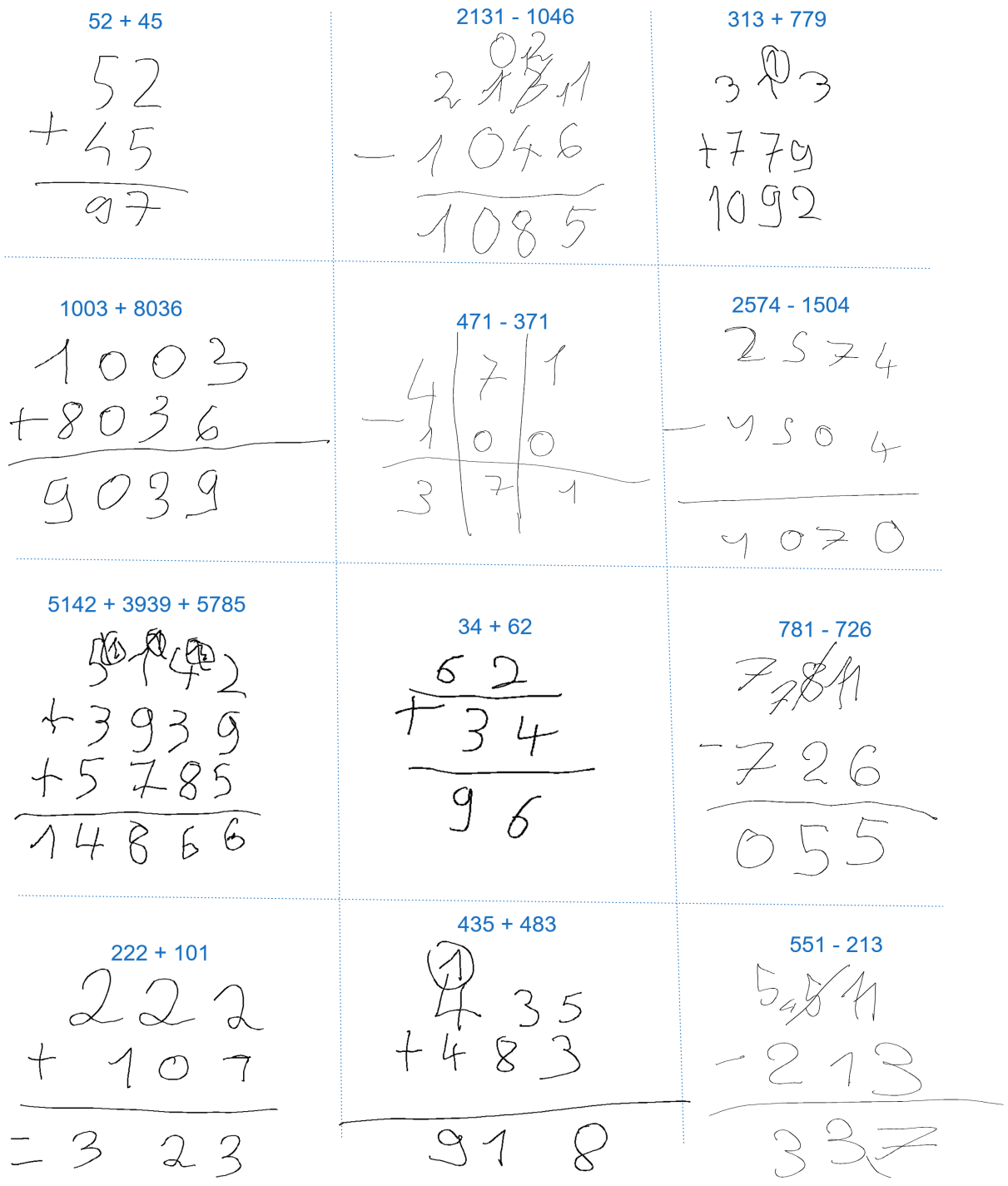

FIGURE 5.1 – Un ensemble d'opérations arithmétiques saisies par des élèves de primaire. Ces opérations sont toutes correctement saisies : à savoir la pose de l'opération et le résultat sont ceux attendus. On remarque cependant que certaines opérations contiennent des tracés non attendus, mais ne rendant pas faux l'opération.

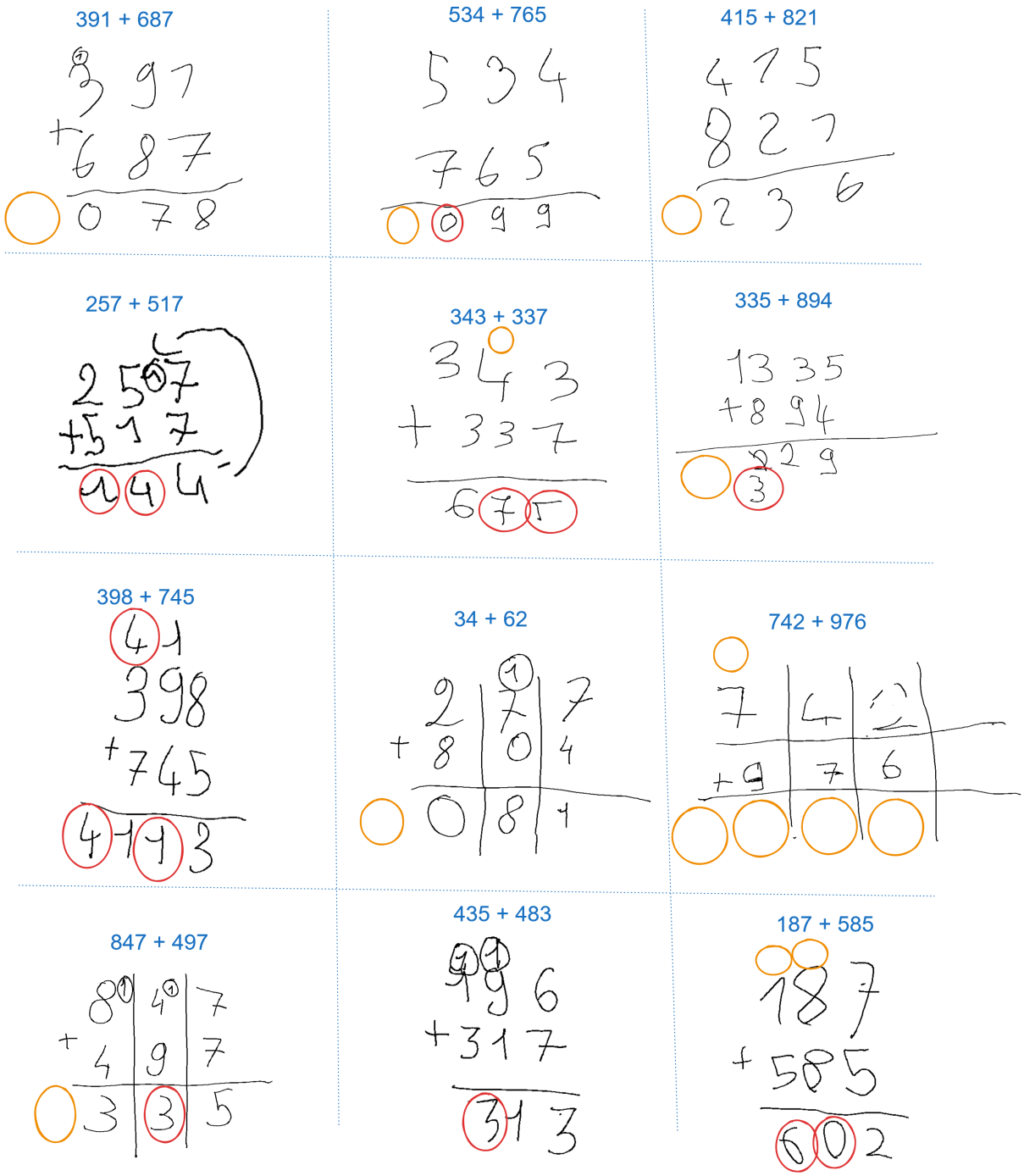

FIGURE 5.2 – Un ensemble d'additions posées saisies par des élèves de primaire. Ces additions contiennent chacune une ou plusieurs erreurs, celles-ci sont mises en évidence : en rouge un symbole dont la valeur est incorrecte, en orange un symbole manquant. On constate ainsi des retenues ou résultats manquants et des erreurs de calculs. Les chiffres non attendus n'intervenant pas dans l'opération ne sont pas considérés comme des erreurs.

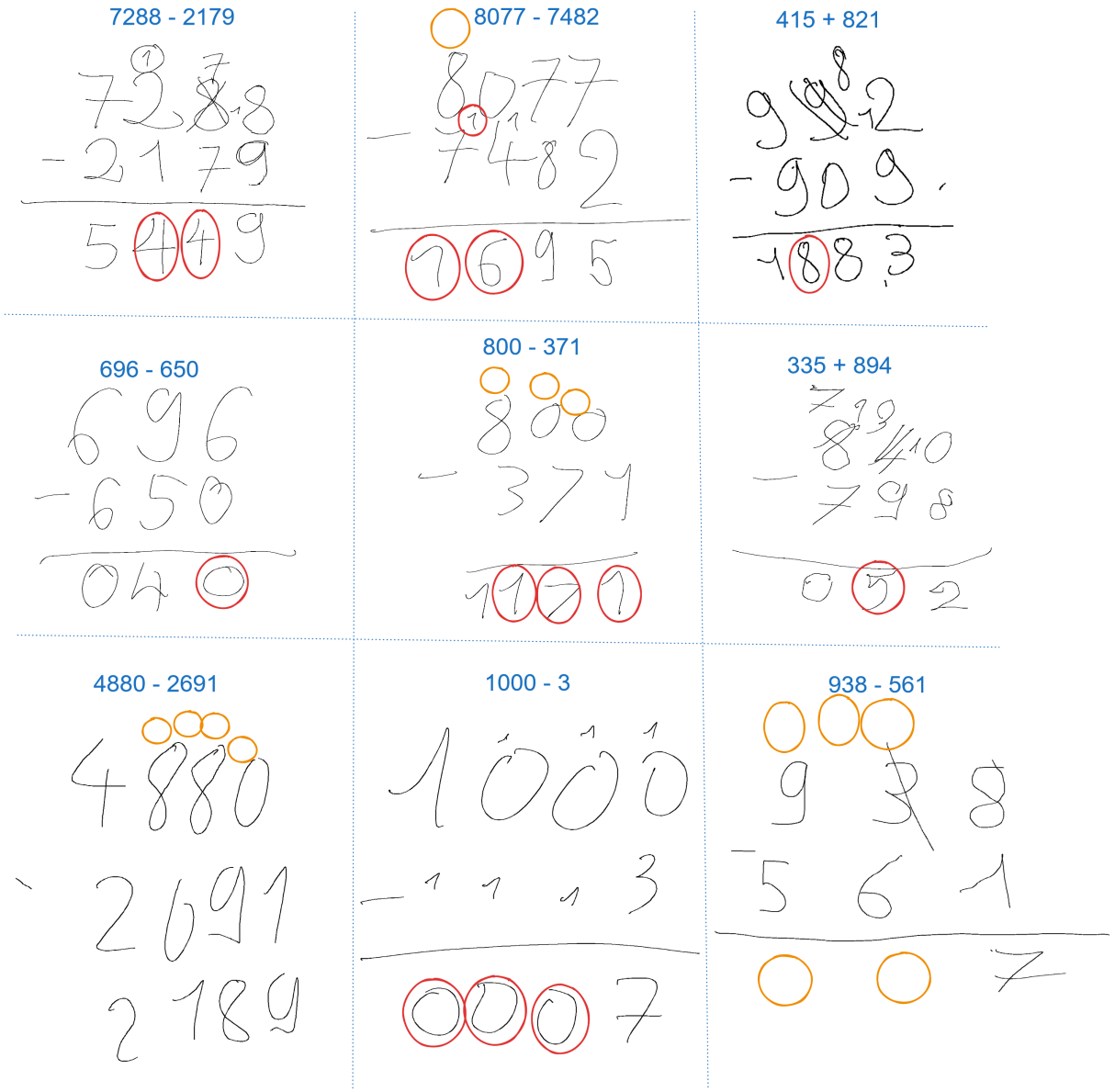

FIGURE 5.3 – Un ensemble de soustractions posées saisies par des élèves de primaire. Ces soustractions contiennent chacune une ou plusieurs erreurs, celles-ci sont mises en évidence : en rouge un symbole dont la valeur est incorrecte, en orange un symbole manquant. On constate ainsi des retenues ou résultats manquants et des erreurs de calculs. Les chiffres non attendus n'intervenant pas dans l'opération ne sont pas considérés comme des erreurs.

PUBLICATIONS

[LAM19a] Arnaud LODS, Eric ANQUETIL et Sébastien MACÉ, « Fuzzy Visibility Graph for Structural Analysis of Online Handwritten Mathematical Expressions », in : *15th IAPR International Conference on Document Analysis and Recognition (ICDAR2019)*, Sydney, Australia, sept. 2019, p. 641-646, URL : https://hal.science/hal-02281462.

[LAM19b] Arnaud LODS, Eric ANQUETIL et Sébastien MACÉ, « Graph matching over Hypothesis Graphs for the Analysis of Handwritten Arithmetic Operations », in : *13th IAPR International Workshop on Graphics Recognition (GREC 2019)*, Sydney, Australia, sept. 2019, URL : https://hal.science/hal-02281511.

[LAM20] Arnaud LODS, Eric ANQUETIL et Sébastien MACÉ, « Graph Edit Distance for the analysis of children's on-line handwritten arithmetical operations », in : *17th International Conference on Frontiers in Handwriting Recognition*, Dortmund, Germany, sept. 2020, URL : https://hal.science/hal-02931889.

[LAM21] Arnaud LODS, Eric ANQUETIL et Sébastien MACÉ, « Segmentation and graph matching for online analysis of student arithmetic operations », in : *16th International Conference on Document Analysis and Recognition*, Lausanne, Switzerland, sept. 2021, URL : https://hal.science/hal-03259438.

BIBLIOGRAPHIE

[Abu+15] Zeina ABU-AISHEH et al., « An Exact Graph Edit Distance Algorithm for Solving Pattern Recognition Problems », in : *4th International Conference on Pattern Recognition Applications and Methods 2015*, Lisbon, Portugal, jan. 2015, DOI : 10.5220/0005209202710278, URL : https://hal.archives-ouvertes.fr/hal-01168816.

[Abu+17] Zeina ABU-AISHEH et al., « Graph edit distance contest : Results and future challenges », in : *Pattern Recognition Letters* 100 (2017), p. 96-103.

[AMV14] Ahmad-Montaser AWAL, Harold MOUCHERE et Christian VIARD-GAUDIN, « A global learning approach for an online handwritten mathematical expression recognition system », in : *Pattern Recognition Letters* 35 (2014), p. 68-77.

[Ant+08] Lisa ANTHONY et al., « Developing handwriting-based Intelligent Tutors to enhance mathematics learning », in : *Unpublished doctoral dissertation, Carnegie Mellon University, USA* (2008).

[ARW11] Ivon ARROYO, James M ROYER et Beverly P WOOLF, « Using an intelligent tutor and math fluency training to improve math performance », in : *International Journal of Artificial Intelligence in Education* 21.*1-2* (2011), p. 135-152.

[ASB14] Francisco ALVARO, Joan-Andreu SÁNCHEZ et José-Miguel BENEDÍ, « Recognition of on-line handwritten mathematical expressions using 2D stochastic context-free grammars and hidden Markov models », in : *Pattern Recognition Letters* 35 (2014), p. 58-67.

[AYK07] Lisa ANTHONY, Jie YANG et Kenneth R KOEDINGER, « Benefits of handwritten input for students learning algebra equation solving », in : *Frontiers in Artificial Intelligence and Applications* 158 (2007), p. 521.

[AYK12] Lisa ANTHONY, Jie YANG et Kenneth R KOEDINGER, « A paradigm for handwriting-based intelligent tutors », in : *International Journal of Human-Computer Studies* 70.*11* (2012), p. 866-887.

[Bai+19] Yunsheng Bai et al., « Simgnn : A neural network approach to fast graph similarity computation », in : *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, 2019, p. 384-392.

[BG17] David B Blumenthal et Johann Gamper, « Exact computation of graph edit distance for uniform and non-uniform metric edit costs », in : *International Workshop on Graph-Based Representations in Pattern Recognition*, Springer, 2017, p. 211-221.

[BG20] David B. Blumenthal et Johann Gamper, « On the exact computation of the graph edit distance », in : *Pattern Recognition Letters* 134 (2020), Applications of Graph-based Techniques to Pattern Recognition, p. 46-57, ISSN : 0167-8655, DOI : https://doi.org/10.1016/j.patrec.2018.05.002, URL : http://www.sciencedirect.com/science/article/pii/S0167865518301685.

[Blo99] Isabelle Bloch, « Fuzzy relative position between objects in image processing : a morphological approach », in : *IEEE transactions on pattern analysis and machine intelligence* 21.7 (1999), p. 657-664.

[Blu+20] David B Blumenthal et al., « Comparing heuristics for graph edit distance computation », in : *The VLDB journal* 29.1 (2020), p. 419-458.

[Bou+17] Sebastien Bougleux et al., « Graph edit distance as a quadratic assignment problem », in : *Pattern Recognition Letters* 87 (2017), p. 38-46.

[Che+19] Xiaoyang Chen et al., « An efficient algorithm for graph edit distance computation », in : *Knowledge-Based Systems* 163 (2019), p. 762-775.

[CLW18] Tyne Crow, Andrew Luxton-Reilly et Burkhard Wuensche, « Intelligent tutoring systems for programming education : a systematic review », in : *Proceedings of the 20th Australasian Computing Education Conference*, 2018, p. 53-62.

[DA11] Adrien Delaye et Eric Anquetil, « Fuzzy relative positioning templates for symbol recognition », in : *2011 12th International Conference on Document Analysis and Recognition (ICDAR)*, IEEE, 2011, p. 1220-1224.

[DA13] Adrien Delaye et Eric Anquetil, « HBF49 feature set : A first unified baseline for online symbol recognition », in : *Pattern Recognition* 46.1 (2013), p. 117-130.

[DA14] Adrien Delaye et Eric Anquetil, « Learning of fuzzy spatial relations between handwritten patterns », in : *International Journal of Data Mining, Modelling and Management 2* 6.*2* (2014), p. 127-147.

[Den+17] Yuntian Deng et al., « Image-to-markup generation with coarse-to-fine attention », in : *International Conference on Machine Learning*, PMLR, 2017, p. 980-989.

[Dih+04] Roberta E Dihoff et al., « Provision of feedback during preparation for academic testing : Learning is enhanced by immediate but not delayed feedback », in : *The Psychological Record* 54.*2* (2004), p. 207-231.

[ES01] Yuko Eto et Masakazu Suzuki, « Mathematical formula recognition using virtual link network », in : *Proceedings of sixth international conference on document analysis and recognition*, IEEE, 2001, p. 762-767.

[Fis+14] Andreas Fischer et al., « A hausdorff heuristic for efficient computation of graph edit distance », in : *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, Springer, 2014, p. 83-92.

[Gao+10] Xinbo Gao et al., « A survey of graph edit distance », in : *Pattern Analysis and applications* 13 (2010), p. 113-129.

[GH16] Karam Gouda et Mosab Hassaan, « CSI_GED : An efficient approach for graph edit similarity computation », in : *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, IEEE, 2016, p. 265-276.

[GLA12] Achraf Ghorbel, Aurélie Lemaitre et Eric Anquetil, « Competitive hybrid exploration for off-line sketches structure recognition », in : *2012 International Conference on Frontiers in Handwriting Recognition*, IEEE, 2012, p. 571-576.

[GML08] Martin Graff, Peter Mayer et Morena Lebens, « Evaluating a web based intelligent tutoring system for mathematics at German lower secondary schools », in : *Education and Information Technologies* 13.*3* (2008), p. 221-230.

[HBT96] Jianying Hu, Michael K Brown et William Turin, « HMM based online handwriting recognition », in : *IEEE Transactions on pattern analysis and machine intelligence* 18.*10* (1996), p. 1039-1045.

[Hua+16] Xudong Huang et al., « Intelligent tutoring systems work as a math gap reducer in 6th grade after-school program », in : t. 47, Elsevier, 2016, p. 258-265.

[HZ13] Lei Hu et Richard Zanibbi, « Segmenting handwritten math symbols using adaboost and multi-scale shape context features », in : *2013 12th International Conference on Document Analysis and Recognition*, IEEE, 2013, p. 1180-1184.

[HZ16a] Lei Hu et Richard Zanibbi, « Line-of-sight stroke graphs and parzen shape context features for handwritten math formula representation and symbol segmentation », in : *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, IEEE, 2016, p. 180-186.

[HZ16b] Lei Hu et Richard Zanibbi, « MST-based visual parsing of online handwritten mathematical expressions », in : *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, IEEE, 2016, p. 337-342.

[JH06] Derek Justice et Alfred Hero, « A binary linear programming formulation of the graph edit distance », in : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.*8* (2006), p. 1200-1214.

[Jul+15] Frank Julca-Aguilar et al., « Top-down online handwritten mathematical expression parsing with graph grammar », in : *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications : 20th Iberoamerican Congress, CIARP 2015, Montevideo, Uruguay, November 9-12, 2015, Proceedings 20*, Springer, 2015, p. 444-451.

[KF16] James A Kulik et JD Fletcher, « Effectiveness of intelligent tutoring systems : a meta-analytic review », in : *Review of educational research* 86.*1* (2016), p. 42-78.

[Kri20] Omar Krichen, « Conception d'un système tutoriel intelligent orienté stylet pour l'apprentissage de la géométrie basé sur une interprétation à la volée de la production manuscrite de figures », thèse de doct., INSA RENNES, 2020.

[Ler+17] Julien Lerouge et al., « New binary linear programming formulation to compute the graph edit distance », in : *Pattern Recognition* 72 (2017), p. 254-265.

[LIN19] Anh Duc Le, Bipin Indurkhya et Masaki Nakagawa, « Pattern generation strategies for improving recognition of handwritten mathematical expressions », in : *Pattern Recognition Letters* 128 (2019), p. 255-262.

[LM10] Qing Li et Xin Ma, « A meta-analysis of the effects of computer technology on school students' mathematics learning », in : *Educational Psychology Review* 22.*3* (2010), p. 215-243.

[LN16] Anh Duc Le et Masaki Nakagawa, « A system for recognizing online handwritten mathematical expressions by using improved structural analysis », in : *International Journal on Document Analysis and Recognition (IJDAR)* 19 (2016), p. 305-319.

[LVN14] Anh Duc Le, Truyen Van Phan et Masaki Nakagawa, « A system for recognizing online handwritten mathematical expressions and improvement of structure analysis », in : *2014 11th IAPR International Workshop on Document Analysis Systems*, IEEE, 2014, p. 51-55.

[Mah+19] M. Mahdavi et al., « ICDAR 2019 CROHME + TFD : Competition on Recognition of Handwritten Mathematical Expressions and Typeset Formula Detection », in : *2019 International Conference on Document Analysis and Recognition (ICDAR)*, 2019, p. 1533-1538.

[Mar+17] Victor J Marin et al., « Automated personalized feedback in introductory Java programming MOOCs », in : *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, IEEE, 2017, p. 1259-1270.

[Mat99] Nicholas Elias Matsakis, « Recognition of handwritten mathematical expressions », thèse de doct., Massachusetts Institute of Technology, 1999.

[Mou+14] Harold Mouchere et al., « ICFHR 2014 competition on recognition of online handwritten mathematical expressions (CROHME 2014) », in : *2014 14th International Conference on Frontiers in Handwriting Recognition*, IEEE, 2014, p. 791-796.

[Mou+16] Harold Mouchère et al., « ICFHR2016 CROHME : Competition on recognition of online handwritten mathematical expressions », in : *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, IEEE, 2016, p. 607-612.

[NMB10] Roger Nkambou, Riichiro Mizoguchi et Jacqueline Bourdeau, *Advances in intelligent tutoring systems*, t. 308, Springer Science & Business Media, 2010.

[NRB06] Michel Neuhaus, Kaspar Riesen et Horst Bunke, « Fast suboptimal algorithms for the computation of graph edit distance », in : *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, Springer, 2006, p. 163-172.

[OS04] Luiz S Oliveira et Robert Sabourin, « Support vector machines for handwritten numerical string recognition », in : *Ninth international workshop on frontiers in handwriting recognition*, IEEE, 2004, p. 39-44.

[PSS18] Ray S Perez, Anna Skinner et Robert A Sottilare, « –A REVIEW OF INTELLIGENT TUTORING SYSTEMS FOR SCIENCE TECHNOLOGY ENGINEERING AND MATHEMATICS (STEM) », in : *Assessment of Intelligent Tutoring Systems Technologies and Opportunities* (2018), p. 1.

[Py01] Dominique Py, « Environnements interactifs d'apprentissage et démonstration en géométrie », in : *Habilitationa Diriger des Recherches, Université de Rennes* 1 (2001).

[RFB07] Kaspar Riesen, Stefan Fankhauser et Horst Bunke, « Speeding Up Graph Edit Distance Computation with a Bipartite Heuristic. », in : *MLG*, 2007, p. 21-24.

[RPA16] Irwansyah Ramadhan, Bedy Purnama et Said Al Faraby, « Convolutional neural networks applied to handwritten mathematical symbols classification », in : *2016 4th international conference on information and communication technology (ICoICT)*, IEEE, 2016, p. 1-4.

[SC13] Saiying Steenbergen-Hu et Harris Cooper, « A meta-analysis of the effectiveness of intelligent tutoring systems on K–12 students' mathematical learning. », in : *Journal of educational psychology* 105.*4* (2013), p. 970.

[Ser14] Francesc Serratosa, « Fast computation of bipartite graph matching », in : *Pattern Recognition Letters* 45 (2014), p. 244-250.

[SKC15] Fotini Simistira, Vassilis Katsouros et George Carayannis, « Recognition of online handwritten mathematical formulas using probabilistic SVMs and stochastic context free grammars », in : *Pattern Recognition Letters* 53 (2015), p. 85-92.

[SZ14] Karen Simonyan et Andrew Zisserman, « Very deep convolutional networks for large-scale image recognition », in : *arXiv preprint arXiv :1409.1556* (2014).

[VHH08] Ba-Quy Vuong, Siu-Cheung Hui et Yulan He, « Progressive structural analysis for dynamic recognition of on-line handwritten mathematical expressions », in : *Pattern Recognition Letters* 29.*5* (2008), p. 647-655.

[Xin+17] Yan Ping Xin et al., « An intelligent tutor-assisted mathematics intervention program for students with learning difficulties », in : *Learning Disability Quarterly* 40.*1* (2017), p. 4-16.

[Yam+06] Ryo Yamamoto et al., « On-line recognition of handwritten mathematical expressions based on stroke-based stochastic context-free grammar », in : *Tenth international workshop on frontiers in handwriting recognition*, Suvisoft, 2006.

[You67] Daniel H Younger, « Recognition and parsing of context-free languages in time n3 », in : *Information and control* 10.*2* (1967), p. 189-208.

[ZBC02] Richard Zanibbi, Dorothea Blostein et James R. Cordy, « Recognizing mathematical expressions using tree transformation », in : *IEEE Transactions on pattern analysis and machine intelligence* 24.*11* (2002), p. 1455-1467.

[ZDD18a] Jianshu Zhang, Jun Du et Lirong Dai, « Multi-scale attention with dense encoder for handwritten mathematical expression recognition », in : *2018 24th international conference on pattern recognition (ICPR)*, IEEE, 2018, p. 2245-2250.

[ZDD18b] Jianshu Zhang, Jun Du et Lirong Dai, « Track, attend, and parse (tap) : An end-to-end framework for online handwritten mathematical expression recognition », in : *IEEE Transactions on Multimedia* 21.*1* (2018), p. 221-233.

[Zha+17] Jianshu Zhang et al., « Watch, attend and parse : An end-to-end neural network based approach to handwritten mathematical expression recognition », in : *Pattern Recognition* 71 (2017), p. 196-206.

[ZJ17] Bilan Zhang et Jiyou Jia, « Evaluating an intelligent tutoring system for personalized math teaching », in : *2017 International Symposium on Educational Technology (ISET)*, IEEE, 2017, p. 126-130.

[ZMV17] Ting Zhang, Harold Mouchere et Christian Viard-Gaudin, « Tree-based BLSTM for mathematical expression recognition », in : *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, t. 1, IEEE, 2017, p. 914-919.

[ZMV18] Ting Zhang, Harold Mouchère et Christian Viard-Gaudin, « A tree-BLSTM-based recognition system for online handwritten mathematical expressions », in : *Neural Computing and Applications* (2018), p. 1-20.

[ZZR21] Dmytro Zhelezniakov, Viktor Zaytsev et Olga Radyvonenko, « Online Handwritten Mathematical Expression Recognition and Applications : A Survey », in : *IEEE Access* 9 (2021), p. 38352-38373.

COLLEGE MATHS, TELECOMS
DOCTORAL INFORMATIQUE, SIGNAL
BRETAGNE SYSTEMES, ELECTRONIQUE

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Titre : Analyse en-ligne de tracés manuscrits d'opérations arithmétiques posées pour l'aide à l'apprentissage sur tablette numérique

Mot clés : opérations arithmétiques, graphes de visibilité floue, correspondance de graphes

Résumé : Cette thèse s'inscrit dans la continuité du LabCom «Script&Labs » qui porte sur l'apport de l'IA pour l'apprentissage numérique. Nous traitons la problématique de l'analyse d'opérations arithmétiques posées manuscrites réalisées sur tablette stylet. Les contributions apportées dans cette thèse portent sur la reconnaissance à la volée d'opérations manuscrites en deux dimensions et sur la mise en correspondance de la réalisation de l'élève avec une consigne donnée afin d'identifier les erreurs de l'élève pour apporter un ensemble de feedbacks personnalisés. Le système se base sur des graphes de visibilité floue pour modéliser les opérations et qualifier les relations entre symboles. Grâce aux caractéristiques floues extraites des opérations, on apparie les symboles avec la représentation d'une solution produite à partir de la consigne. En proposant un appariement itératif sur une segmentation d'opérations en graphes de nombres, nous avons abouti à un résultat d'analyse qualitatif avec des temps de traitement garantissant des retours immédiats. Les expérimentations ont été conduites sur une base de données significative de saisie manuscrite d'enfants comportant différentes opérations de tailles variables (additions, soustractions, multiplications). Les résultats obtenus montrent l'impact de nos contributions sur la capacité du système à aboutir en moins de 5 secondes à une analyse correcte des propositions manuscrites des élèves.

Title: On-line analysis of handwritten arithmetical operations for digital learning on numerical tablet

Keywords: arithmetical operations, fuzzy visibility graph, graph matching

Abstract: This thesis is in the context of the LabCom «Script&Labs» which deals with the contribution of AI in Digital Learning, here to solve the problem of arithmetical operations analysis written on tablet. The contributions of this work are brought for the on 2D operations' recognition on the fly and the matching between the recognition result and a given instruction in order to understand the student solution and to give him personalized feedbacks. The system is based on fuzzy visibility graph built to represent the operations and qualify relationship between mathematical symbols. Using fuzzy characteristics extracted, the system match recognized symbols with the expected answer representation built from the instruction. By building an iterative matching strategy based on a new graph of lines segmentation, we achieve the analysis with processing time guarantying immediate feedbacks to the student. Experiments were conducted on a significant database of student handwritten arithmetical operations which contains different types of varying sizes (additions, substractions, multiplications). Results show the impact of our contributions to achieve in less than 5s a valid analysis of handwritten student solutions.