

למידה עמוקה - Deep Learning

תזכורת יש לנו שני סוגים מרכזיים של למידה מדוגמאות:

- Supervised Learning - דוגמאות מתוייגות
- Unsupervised Learning - דוגמאות לא מתוייגות

Machine Learning, בסופו של דבר, לומד פונקציה $g(\vec{x})$ שמקבלת ווקטור ומחזירה ערך (בדיד או רציף). אם היינו יכולים לכתוב את הפונקציה בצורה מפורשת היינו עושים את זה (בדרך כלל ע"י Decision Tree), אבל בהרבה מקרים זה לא עובד - המומחים לא מצליחים לתאר בצורה מפורשת את הפונקציה, לא יודעים לכתוב את עץ ההחלטות בצורה טובה, אבל אנשים כן יודעים לתת דוגמאות, ולכן מנסים ללמוד את הפונקציה מתוך הדוגמאות.

למידה עמוקה מאוד טובה בתמונות¹, ומצליחה ללמוד באמצעות פחות דוגמאות משיטות אחרות (למרות שעדיין צריך כמות מאוד גדולה של דוגמאות והרבה כוח חישוב). היא טובה גם בדיבור. למידה עמוקה לא מצליחה בטקסט.

מה זה בעצם Deep Learning?

למידה עמוקה משתמשת בNeural Networks עם הרבה שכבות חבויות (hidden layers). אלגוריתמים קודמים (backpropagation) לא הצליחו להתכנס עם יותר מ2 שכבות.

תזכורת: יחידת חישוב ברשת ניורונית מקבלת כמה קלטים (משכבת הקלט או משכבות חבויות קודמות) ומחשבת סכום משוקלל שלהם. את הערך הסופי מכניסים לפונקציית הסיגמואיד $f(x) = \frac{1}{1+e^{-x}}$, שנותנת ערכים פחות-או-יותר דיסקרטים והיא גזירה (ולכן אפשר לכנס את הרשת).

הבעיה היא שהרבה פעמים הלמידה לא מתכנסת. אחת הסיבות היא שכשעושים back propogation קשה לדעת מה מקור הטעות, איזה משקלים צריך לעדכן.

בלמידה עמוקה משתמשים ב $f(x)$ לא-מונוטונית, שמסוגלת לחלק את הדוגמאות בצורה יותר חופשית. יש גם אלגוריתם בשם SVM שמשמש ב $f(x)$ לינארית אבל מבצע הטלה של הדוגמאות על מרחב אחר כדי שיהיה אפשר לעשות חלוקה לינארית).

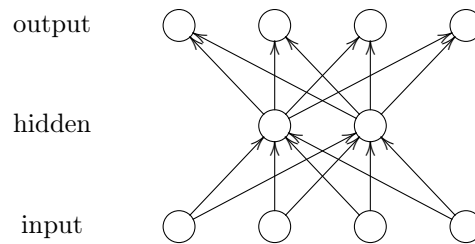
למה multilayers עובד?

כאשר ניסו לנתח רשת ניורונית שלמדה לזהות כתב, גילו שהשכבות הפנימיות מתחלקות לnode's, כאשר כל node למד אזור מסוים של תמונת הקלט, והשכבות הבאות שילבו את המידע הזה.

¹היו גם נסיונות מוצלחים להשתמש בDeep Learning בשביל לשחק גו - למרות ששם היתה הרכבה של הרבה שיטות.

אז מה Deep Learning עושה?

- משתמש במליוני דוגמאות (ודורש כוח חישוב עצום)
- $f(x)$ משתמש בפונקציית Relu במקום בסיגמואיד. Relu עושה $\max(x, 0)$. היא פחות טובה לגזירה אבל יותר להתכנסות.
- בכל פעם שהרשת לומדת, היא מכבה באופן רנדומלי 50% מכל שכבה (לא מאפשרים להם להשתנות). זה מונע מהמערכת ללמוד את עצמה (במקום את הדוגמאות).
- שיטה שלא כולם משתמשים בה - autoencoding - אימון כל שכבה בנפרד (במקום כל השכבות בבת אחת):
 1. ה input layer וה output layer צריכים להיות באותו גודל.
 2. בונים שכבה חבויה אחת שיותר קטנה ברוחב מהרמה שאליה נכנסנו.
 3. מצפים שהערכים יהיו אותו דבר. זה אומר שהשכבה החבויה הצרה יותר תמצא אוטומטית features שמתארים את הנתונים:



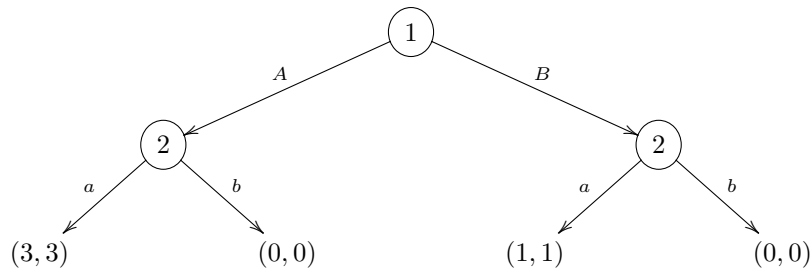
תורת המשחקים - מה קורה כשיש יריב?

המשחקים שלמדנו בעבר היו משני סוגים, שניהם full information:

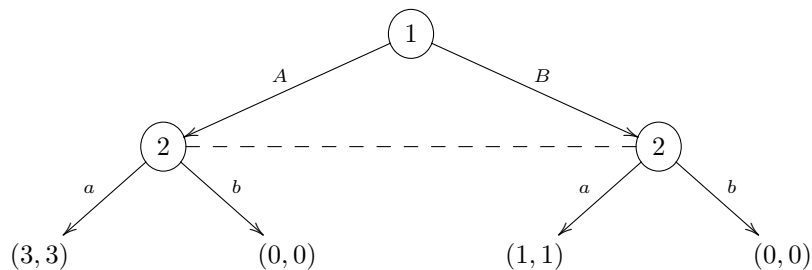
- משחקים שמיוצגים באמצעות מטריצות (Normal Form Games), שחקן אחד על השורות והשני על העמודות, וכל אחד בוחר פעולה והתא במטריצה מייצג את התמורה של שניהם. כאן בכל שלב ידענו מה מצב העולם, אבל היה imperfect information כי לא ידענו מה השחקן השני הולך לעשות.
- משחקים שניתן לייצג באמצעות עץ - perfect information. למשל שח. מצב העולם ידוע - יודעים מה השחקן השני עשה קודם.

Incomplete Information

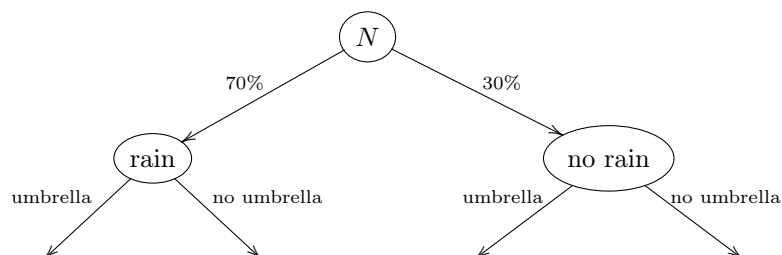
מה קורה אם אין full information? יכול להיות שאנחנו לא יודעים באיזה משחק אנחנו משחקים - יש הסתברות לגבי התמורות. את המשחקים האלה אפשר לתאר באמצעות גרף, כאשר קודקוד בגרף הוא נקודת החלטה של אחד השחקנים:



לגרף הזה מוסיפים information sets - קבוצות של קודקודים שהשחקן לא יודע באיזה מהם הוא נמצא - וככה אפשר לתאר imperfect information:



כדי לתאר הסתברות מוסיפים שחקן חדש - N (טבע - Nature):



אם אנחנו לא יודעים מה הטבע בחר - אז צריך להוסיף information set, ואז יש לנו incomplete information:

