Jeopardy- How Watson Played It

Jeopardy! :

Jeopardy! A highly popular TV quiz show that originated in the US, and now airs in dozens of international markets. Jeopardy! relies on many human cognitive abilities traditionally seen beyond the capability of computers. The show features challenging questions (called “clues” in the show’s dialect) drawn from a very broad array of topics; the questions embody all manner of complex and ambiguous language, including all the vague allusions and hints, irony, humour and wordplay. The material for the questions covers a wide variety of topics (including history and current events, the sciences, the arts, popular culture, literature, and languages) and there are also pun-laden titles (many of which refer to the standard subjects), wordplay categories. The general rules of game play are as follows. There are two main rounds of play, wherein each round uses a board containing 30 squares, organized as five squares in six different categories, with each square containing a hidden question. Second-round questions have higher dollar values, presumably reflecting greater difficulty.

In typical play, a square will be selected according to category and dollar amount by the player in control of the board, and its question will be revealed and read aloud by the host. When the host finishes reading the question, players may attempt to answer, i.e., “buzz in,” by pressing a signalling device. The first player to buzz in obtains the right to attempt to answer; if the answer is correct, the player’s score increases by the question’s dollar value, whereas if the answer is incorrect, the player’s score decreases by the dollar value, and the question is re-opened for the other players to attempt to answer on the “rebound.” One question in the first round, and two questions in the second round have a special “Daily Double (DD)” status. The player who selected the question has the exclusive right to answer, and player must specify a wager between $5 and either her current score or the round limit, whichever is greater. The game concludes with a single question in the “Final Jeopardy” round. The players write down sealed-bid wagers, and then have 30 seconds to write an answer after the question is revealed. The player finishing with the highest dollar-value score wins that amount, and can continue playing on the next episode.

One might think and relate Jeopardy! to the popular TV show in India, Kaun Banega Crorepati (KBC)[1]. These two gameplays can be assumed to be oppositely oriented. KBC is a single player game, presumably easier than Jeopardy. KBC is a simple question-answering game, each question having four options with single correct answer whereas the Jeopardy is similar to predicting a question to answer (the so called question). The game also includes four helplines namely audience poll, phone a friend, double tip and ask an expert which makes the game easier. Jeopardy doesn’t feature all these helplines.

Jeopardy gameplay approaches and techniques:

Generally each player does have a different gameplay strategy. It is apparent that the players should have a wide range of knowledge to answer the questions and also the players should have a great command over the English language to understand discern double meanings of words, puns, rhymes and inferred hints. But there are certain strategies while decision making which help the humans in increasing their winning probabilities. There are four main areas in the game where different strategies can lead to different winning probabilities.

- Deciding whether to attempt to answer
- Selecting the next square when in control of the board
- Daily Double wagering
- Final Jeopardy! Wagering

Most of the contestants have a habit of selecting in top-to-bottom order within a given category and to stay within a category rather than jumping across categories. There is a further weak tendency to select categories moving left-to-right across the board. On the basis of these observations.This is a question-selection strategy, employed during the Jeopardy! round or the Double Jeopardy! round, in which the next question is selected from a randomly-chosen category different from the category of the last question, potentially taking an advantage with control of the board by confusing his or her opponents. Also given importance to DD, player's square selection strategy results in a high likelihood of finding a Daily Double. From the analysis of the previous game shows a player can conclude that DD's are more probable to occur in 3rd and 4th rows rather than in 1st and 2nd rows.

There are several wagering rule of thumb strategies depending on the player's position in the game play. A few are mentioned below[2]:

- **Falk's Law:**
  A pessimistic adage stating that the likelihood of supplying the correct response to a Daily Double or to the Final Jeopardy! question is inversely proportional to the percentage of Forrest Bounce.
- **Bridges's Rule**:
  A wagering "rule of thumb" that suggests a player never bet to tie in Final Jeopardy! in tournament play, except if betting to win risks a loss that betting to tie would not risk.
- **Clavin's Rule:**
  A wagering "rule of thumb" that suggests a player not put a lock game at risk with an excessive wager, however tempting the category.
- **Crush:**
  In Final Jeopardy! wagering strategy, the scenario in which the score of the player in second place is between one-half and two-thirds of the score of the player in first place. Under this scenario, given rational wagers by the players, a win is possible for the second-place player only if the first-place player misses Final Jeopardy! and the second-place player gets it right. Here, the first-place player should bet enough so as to have at least one dollar more than the second-place player if the second-place player bets it all, and the second-place player should bet it all, since he or she must give the correct response to win anyway, unless he or she has a lock for second place and would prefer to limit his or her wager to protect that position.
- **Faith Love:**
  In Final Jeopardy! wagering strategy, the scenario in which the 1st-place player's score is equal to twice the difference between the scores of the 2nd- and 3rd-place players. Under this scenario, the 1st-place player should bet $0, because the 2nd-place contestant has reason to only bet to tie, since the bet to keep 3rd locked out is identical to the bet that would tie the leader if 2nd is correct.
- **Jeeks's Rule:**
  A wagering "rule of thumb" that suggests that in a two- or three-way tie going into Final Jeopardy! a player should wager either all or nothing.

- In Final Jeopardy! wagering strategy, the scenario in which the score of the player in second place is less than one-half the score of the player in first place. Under this scenario, the leader is assured a victory so long as he or she does not wager more than (leader's score - 2*(second place score)), or (leader's score - 2*(second place score) - $1) to prevent the possibility of a tie.
- **Lock-tie:**
  In Final Jeopardy! wagering strategy, the scenario in which the score of the player in second place is exactly one-half the score of the player in first place. Under this scenario, the leader is assured a victory or a tie victory so long as he or she wagers $0; the second-place player should go for the tie and wager everything.
- **Prisoner's dilemma:**
  In Final Jeopardy! wagering strategy, the scenario in which two players are tied for the lead and the third player has less than one-half their scores. Under this scenario, the leaders must each independently decide whether the other will bet $0 or everything, and then bet likewise. Betting everything has the unfortunate outcome of a double loss in the case of a double stumper: $0 is never a winning score on Jeopardy! The third-place player should risk nothing in this scenario.

If the player has to consider the above wagering strategies, in order to gain a maximum advantage during wagering. Reducing the physical errors adds up the winning chances to the player.

Watson Computer:

- **Architecture of the computer:**

  Watson is a huge supercomputing machine. Watson is made up of a cluster of ninety IBM Power 750 servers [3] (plus additional I/O, network and cluster controller nodes in 10 racks) with a total of 2880 POWER7 processor cores and 16 Terabytes of RAM. Each Power 750 server uses a 3.5 GHz POWER7 eight core processor, with four threads per core. Watson is made up of a cluster of ninety IBM Power 750 servers (plus additional I/O, network and cluster controller nodes in 10 racks) with a total of 2880 POWER7 processor cores and 16 Terabytes of RAM. Each Power 750 server uses a 3.5 GHz POWER7 eight core processor, with four threads per core. Watson has a processing power of 500 gigabytes the equivalent of a million books, per second. The information is stored in Watson's RAM for the game because data stored on hard drives is too slow to access.

  For Watson to be able to compete with the humans it should have

  - The ability to discern double meanings of words, puns, rhymes, and inferred hints.
  - Extremely rapid responses.
  - The ability to process vast amounts of information to make complex and subtle logical connections

  In a human, these capabilities come from a lifetime of participation in human interaction and decision-making along with an immersion in pop culture.
  Hence the IBM research people focused mainly on these three capabilities:

  1. Natural Language processing[4]
  2. Hypothesis Generation
  3. Evidence based Learning

Human language is ambiguous, contextual and implicit grounded only in human cognition. There will be seemingly infinite ways to express the same meaning. Watson is developed to meet all these standards to play the game.
Watson’s approach to a particular answer involves 4 stages.

Stage 1: Question Analysis:

In Question analysis Watson’s parsing and semantic analysis capabilities like a deep Slot Grammar parser, a named entity recognizer, a co-reference resolution component, and a relation extraction component are used. A numerous detection rules and classifiers using features from this analysis are being applied to detect critical elements of the question, including:

1) The part of the question that is a reference to the answer (the focus)
2) Terms in the question that indicate what type of entity is being asked for (lexical answer types)
3) A classification of the question into one or more of several broad types
4) Elements of the question that play particular roles that may require special handling.

Question analysis section receives as input the unstructured text question and identifies syntactic and semantic elements of the question, which are encoded as structured information that is later used by the other components of Watson.
The focus of the question is the part of the question that, if replaced by the answer, makes the question a stand-alone statement. The focus is used, for example, by algorithms that attempt to align the question with a potential supporting passage; for proper alignment, the answer in the passage should align with the focus in the question. LATs are terms in the question that indicate what type of entity is being asked for. The headword of the focus is generally a LAT, but questions often contain additional LATs, and in the Jeopardy! domain, categories are an additional source of LATs. LATs are used by Watson’s type coercion components to determine whether a candidate answer is an instance of the answer types. Question Classification identifies the question as belonging to one or more of several broad types. The Question Classes (QClasses) are used to tune the question-answering process by invoking different answering techniques, different machine learning models, or both. QSections are question fragments whose interpretation requires special handling. Some of the most important uses of QSections are to identify lexical constraints on the answer and to decompose a question into multiple sub questions. Like QClasses, QSections are identified either by Prolog rules over the PAS or by regular expressions over the text.

Foundation of question analysis:

Parsing and semantic analysis:

Watson’s parsing and semantic analysis suite composes of the Slot Grammar parser ESG (English Slot Grammar) with an associated Predicate-Argument Structure (PAS) builder, a Named Entity Recognizer (NER), a co-reference resolution component, and a relation extraction component. ESG is used to parse each sentence in a question using a tree structure that shows both surface structure and deep logical structure.

Question analysis adaptations:

On the Jeopardy! show, questions are displayed in all uppercase. This presents challenges for a computer. Mixed case provides information, for example, to indicate proper names or titles. To report this problem, a statistical true-caser component that has been trained on many thousands of correctly cased example phrases has been applied.

Special-purpose relations:

The relation extraction component has been adjusted in two ways to meet the specific characteristics and idiosyncrasies of Jeopardy! questions.

1. Special-purpose relations of particular relevance to Jeopardy! are being identified, and relation detectors for them are developed.
2. Relation detection mechanisms can also be used to identify certain (non-relational) aspects of Jeopardy! questions, which are subsequently used in Question Classification and special question processing.

Focus and LAT detection:

Initially baseline focus and LAT are detected by implementing predefined patterns, in order of their priority, with ties broken by taking the lexically first. There may be mistakes during the baseline focus and LAT detection and also many useful LATs might not have been detected. Hence some mechanisms are implemented to improve the detection of focus and LATs within the question. And then several techniques are implemented to extract the LAT from the category provided. Several LATs will be detected and since they need to be filtered to approach the final answer the LAT confidence is estimated for each LAT using logistic regression classifier and low confidence LATs are filtered. Watson also remembers the LATs from the previous questions in the same category this helps Watson to infer entirely new LATs that would not have been detected by the rules.

Question Classification and QSection detection:

Since Jeopardy! domain contains a huge number of questions, Watson classifies the questions into various categories called Qclasses like definition, puzzle, etymology etc. Watson uses a variety of techniques to identify Qclasses in the question. QSections:
A QSection is an annotation made over a contiguous span of text in the question to represent a function that the text plays in the interpretation of the question. Examples: LexicalConstraints, Abbreviations, SubQuestionSpan etc.

Example question:
POETS & POETRY: He was a bank clerk in the Yukon before he published “Songs of a Sourdough” in 1907.
Focus: ‘He’
LATs: ‘he’, ‘clerk’ and ‘poet’.
Parsing and semantic analysis identify predications publish (e1, he, ‘‘Songs of a Sourdough’’) and in(e2, e1, 1907), the latter saying that publishing event e1 occurred in 1907. It also resolves the co-reference of the two occurrences of “he” and “clerk”, identifies “Yukon” as a geopolitical entity and “Songs of a Sourdough” as a composition, and extracts relations such as authorOf(focus, ‘‘Songs of a Sourdough’’) and temporalLink(publish(...), 1907).

Stage 2: Hypothesis Generation:

Watson reads, analyses and tries to understand millions of books and documents. For each interpretation of the question Watson searches millions of documents to come up with thousands of possible answers. Then it narrows down to few answers. More the number of answers greater the probability of finding the correct answer because it Watson couldn't include the correct answer in the possible answers in the initial stage Watson will never find the correct answer.

Stage 3: Hypothesis and Evidence Scoring:

After downgrading the wrong answers Watson collects passages from many sources to defend positive and negative evidences to the remaining answers. Watson understands the passages having learnt the relationship between the words. Watson scores the algorithms and rates the quality of the evidences based on the information available in the sources and reliability of the sources. Watson runs thousands of algorithms to estimate confidence of each and every answer.

Stage 4: Final Merging and Ranking:

Different types of evidence are better at solving different types of questions. Watson uses its experience for answering similar questions. By playing thousands of practice games Watson learns how to weigh, apply and combine its own algorithms to help decide to the degree to which each piece of evidence is useful or not. These weighted evidence scores are merged together to decide the final ranking for all the possible answers. Watson estimates its confidence as to whether or not for its top answer along with every other possibility is correct. This confidence is based on how high the answer is rated during evidence scoring and ranking. It the Watson's confidence for its top answer is low under 50% then Watson won't answer.

- **Watson Gameplay Strategy:**

The game winning strategies require rapid-fire answers to challenging natural-language questions, broad general knowledge, high precision, and accurate confidence estimates. Strategies are explicitly based on estimating and optimizing a player's probability of winning in any given Jeopardy! game state.

Jeopardy simulation model:

Since Watson should be able to predict the more probable and predicted outcomes rapidly in the gameplay, the models are simulated taking into account the game rules, performance profiles of other players and these models are constantly developed based in the Watson's performances in practice games. There has been a complete analysis on the data of human performance profiles available at J-archive[5]. Using the event data of about 3000 episodes available at J-archive, 3 human models have been devised corresponding to different levels of opposition. The Average contestant model is devised to fit Watson's opponents in former Jeopardy! sparring practice matches. Later a Champion model is devised to represent much stronger opponents who competed in final and semi-final rounds.

Daily Double Betting Model:

Watson is much better in finding the Daily Doubles in the game than human contestants. An analysis of data of all the previous Jeopardy seasons revealed that DD's are most likely to occur in 1st column and least likely to occur in 2nd column and also 2 DD's in the 2nd round never appear in the same row and the column-row frequencies where the DD's are more probable to occur can be calculated. The principle approach to DD betting is based on estimating Watson's likelihood of answering the DD question correctly which is provided by DD Confidence model which is based on thousands of tests on historical categories containing DDs, the model estimates Watson's DD accuracy given the number of previously seen questions in the category that Watson got right and wrong and by estimating how a given bet will impact Watson's overall winning chances if Watson gets the DD right or wrong. A game-state evaluator (GSE)[6] is used to estimate the impact of a bet on the winning chances of Watson. The GSE consists of a nonlinear function approximator that receives a feature-vector description of the current game state and outputs an estimate of the probability that each player will ultimately win the game. GSE enables us to estimate estimated winning chances of a bet, *E*(bet) according to

$$E(bet) = p_{DD} \times V(S_w + bet, \dots) + (1 - p_{DD}) \times V(S_w - bet, \dots)$$

Where *Sw* is Watson's current score, p_{DD} is the in-category confidence, and *V*( ) is the game-state evaluation after Watson's score either increases or decreases by the bet and the DD has been removed from the board. *E*(bet) is calculated for all the legal bets and bet with highest estimated winning chance is selected.

Multigame Wagering model:

In some Jeopardy! contests the winner is determined by their performance in two games to determine first, second and third places. Hence wagering strategies play a very important role. The wagering strategies must vary from game to game and also should be different from single game wagering. Watson is trained with two different neutral networks for both the games 1 and 2. By training the game 2 neutral network[7], the expected probabilities of Watson ending the match in the first, second, or third place, starting from any combination of Game 1 final scores can be estimated by extensive offline Monte Carlo simulations[8].

Final Jeopardy! Wagering:

Final Jeopardy! wagering involves computation of a Best Response strategy[9] to the human Final Jeopardy! model. Best response is computed as follows. Initially Watson's confidence is estimated given the category title. This is done based on samples of Watson's performance in thousands of historical Final Jeopardy! categories, using Natural Language Processing -based feature vector representations of the titles. Second, given Watson's confidence and the human accuracy and correlation parameters, analytic probabilities of the eight possible right or wrong outcomes are derived. Then, for a given Final Jeopardy score combination, on the order of 10,000 Monte Carlo samples of bets from the human models are drawn. Finally, the equity of every legal bet is evaluated, given the human bets and the right/wrong outcome probabilities, and the bet with highest equity is selected.

Square Selection:

Finding a daily double is a very important factor in the game which significantly boosts the game of the player simultaneously decreasing the opportunity to the other players. A simulator is used to systematically investigate the relative importance for Watson in finding the DDs, retaining control of the board if a DD is not found, and learning the crux of a category. These studies are performed using Champion and Grand Champion human models, which feature over DD finding, aggressive DD wagering, and high DD accuracy. If there are any unrevealed DDs, a square i* is selected such that p_{DD}(i) + α*p_{RC}(i) is maximized, where p_{DD}(i) is the probability that square i contains a DD, p_{RC}(i) is an estimated probability that Watson will retain control of the board if i does not contain a DD. The first term in the above equation is calculated using Bayesian inference[10]. The second probability is estimated by combining the simulation model of human performance on regular questions with a model of Watson that adjusts its attempt rate, precision, and buzzability as a function of the number of right/wrong answers previously given in the category. If all the DD's in the round have been found, square is selected with lowest dollar value in the category which indeed helps in increasing the expected accuracy for higher value questions in the same category.

Confidence threshold for attempting the buzz:

Watson attempts to buzz in if and only if the confidence of its top-rated answer exceeds an adjustable threshold value. The threshold is set to a default value that is tuned to maximize expected earnings in most of the game states. Although, at the end of the game, the threshold may vary significantly from the default value. If Watson has a big lead by the end of second round, it is checked whether Watson has a guaranteed lockout by not buzzing on the current square. If so, and if the lockout is no longer guaranteed if Watson buzzes and is wrong, Watson is prohibited from buzzing, regardless of confidence. Exact optimal buzz-in policy for any game state with a question in play can be written as [B_0*(c, D), B_1(c, D), B_2(c, D), B_3(c, D)] given Watson's confidence c and the dollar value D of the current question. The policy is a four-component vector as there are four possible states in which Watson may buzz: the initial state, the first rebound where human #1 answered incorrectly, the first rebound where human #2 answered incorrectly, and the second rebound where both humans answered incorrectly. Dynamic Programming (DP)[11] techniques are used to calculate the optimal policy which involves writing a recursion relation between the value of a current game state with k questions remaining before Final Jeopardy! and values of the possible successor states with k-1 remaining.

$$V_k(s) = \int \rho(c) \sum_{j=1}^{5} p(Dj) \max(\vec{B}\,(c, D_j)) \times \sum_{\delta} p\big(\delta\big|\vec{B}, c\big)\, V_{k-1}(s'(\delta, D_j))dc$$

Probability density of Watson's confidence is denoted by ρ(c), p(D_j) denotes the probability that the next square selected will be in row j with dollar value D_j = 400×j, the max function operates over Watson's possible buzz/no-buzz decisions, p(δ|B,c) denotes the probability of various unit score-change combinations δ, and s' denotes various possible successor states after the D_j square has been played, and a score change combination δ occurred. Exact Dynamic Programming computation of the optimal buzz-in policy involves expanding the root state to all possible successor states, going all the way to Final Jeopardy! states, where the Final Jeopardy! States are evaluated by Monte Carlo trials, and then working backwards using the above equation to compute the optimal buzz policy in the root state. Generally this is too slow to use in live play, since the buzz-in decision must take at most 1-2 seconds. Therefore an Approximate Dynamic Programming calculation is implemented in which the above equation is only used in the first step to evaluate V_k in terms of V_{k-1}, and the V_{k-1} values are based on plain Monte Carlo trials. Since the exact calculation is very slow for higher k values, the accuracy of the Approximate Dynamic Programming cannot be estimated for $9 \leq k < 5$. The Approximate Dynamic Programming usually gives good threshold estimate for $k \leq 5$ remaining squares.

- **Comparing with human game play:**
    1. Watson is highly effective in finding Daily Doubles when compared to humans.
    2. Watson's wagering strategies are way high superior than that of humans because humans cannot perform the precise equity and confidence estimates and complex decision calculations performed by Watson in real time.

3. Also humans are incapable of performing live Bayesian inference calculations of Daily Double location likelihood.
4. Also the Watson’s strategy algorithms exceed human capabilities in real time decision making.
5. Watson’s error rate will be very low when compared to human error rates and also the errors can be corrected using sophisticated algorithms and strategies, for example for a human, a failure to buzz is indistinguishable from losing the buzz, and even if a contestant buzzes and is wrong, the decision may be correct at high enough confidence. Although human buzz errors are small there is a small chance where humans can make major errors, such as mishandling a lock-tie or needlessly locking themselves out.
6. If the questions are small i.e. if very small amount of information is available in the question humans can answer the question easily whereas Watson can find it very difficult since it couldn’t analyse the question properly.
7. Having decent hold on the language, humans are far better in analysing the question than Watson since the human brain is far superior than nlp algorithms run by Watson.

- **The Final Game:**
  The complete game consists of 3 games. The three contestants are Watson, Ken Jennings and Brad Rutter. First game was held on 14th February 2011. Watson could take a commanding lead in Double Jeopardy! Correctly responding to the Daily Doubles. Watson didn’t get everything right displaying a significant number of wrong and low-certainty answers. After round one Watson was tied with Brad for first with $5000, and Ken Jennings was in third with $2000.
  Despite of Watson’s of impressive performance in 2nd game, during Final Jeopardy! Watson made a mistake which embarrassed the IBM Watson research group. By the end of the second game Watson in the lead with a whopping $35,734, Rutter in second place with $10,400 and Jennings bringing up the rear with $4,800. The computer struggled with some categories, like "Actors Who Direct," in the last game where the other two contestants took advantage of this. By the end of first round, Watson was leading with $8600 against Ken’s $4800 and Brad’s $2400. But in the second half of the show, Watson fired up its Deep Question Answer (QA) algorithms and pulled ahead, winning some high-value questions and making only two mistakes. Before Final Jeopardy! Watson is with $23,440, Ken is with $18,200, and Brad is with $5,600. The Final Jeopardy category was "19th Century Novelists," and the question, "William Wilkinson's 'An account of the principalities of Wallachia and Moldavia' inspired this authors's most famous novel." Watson and both humans got it right, answering: "Who is Bram Stoker?” By the end of 3 games, Brad's total score was $21,600. Ken finished just a little ahead with $24,000. Watson finished with a commanding total of $77,147.

  Though Watson won the game, it might be worth to note that if Ken had wagered all he had, he'd have ended up with a final total of $41,200, and if Watson had wagered all it had but got it wrong, its final total would've been $35,734. In this scenario, Ken would have won! Indeed, Watson was very confident in its answer: Watson exactly wagered the exact amount ($17,973) that, even if it had gotten the answer wrong, would keep its score ahead of Ken's; in this case, Watson would end up with $41,201 and Ken with $41,200. A $1 dollar difference!!!
  Appendix:

Natural Language Processing[4]:

Natural language processing refers to the use and capacity of systems to process sentences in a natural languages. The main goal of natural language processing is communication. Natural languages are highly ambiguous. Hence for a computer to understand the language, first it must be disambiguated. So, initially speech is recognized and syntactic, semantic and pragmatic analyses are done. In case of ambiguities all the probable interpretations are made.

Word Segmentation:

A string of characters (graphemes) are broken into a sequence of words.

Morphological Analysis:

Morphology is the field of linguistics which studies the internal structure of words. A morpheme is the smallest linguistic unit that has semantic meaning. Morphological analysis is the task of segmenting a word into its morphemes.

Parts Of Speech tagging:

Each word in a sentence is annotated with a part of speech.

Phrase Chunking:

All non-recursive noun phrases and verb phrases in a sentence are identified.

Syntactic Parsing:

Syntactic parse tree for each sentence is produced.

Semantic Role Labelling:

For each clause, the semantic role played by each noun phrase which is an argument to the verb is determined.

Semantic Parsing:

A semantic parser maps a natural-language sentence to a complete, detailed semantic representation.

Textual Entailment:

Whether one natural language sentence entails (implies) another under an ordinary interpretation is determined.

Then pragmatic analysis is performed and text is summarized and information is extracted.

Monte Carlo Simulations[8]:

Monte Carlo methods are generally a broad class of computational algorithms that basically rely on repeated random sampling to obtain numerical results. These are achieved by running simulations huge number of times over in order to calculate the same probabilities heuristically. Monte Carlo methods are majorly used in 3 distinct problems.

1. Optimization
2. Numerical integration
3. Generation of samples from a probability distribution.

Monte Carlo simulations perform risk analysis by constructing models of conceivable results by substituting a range of values-a probability distribution, for any factor that has inherent uncertainty. It then computes results again and again, each time using a different set of random values from the probability functions. Depending upon the number of uncertainties and the ranges specified for them, a Monte Carlo simulation involves thousands or tens of thousands of recalculations before it completes. Monte Carlo simulation produces distributions of probable outcome values. Monte Carlo simulations provide a large number of advantages over deterministic, or single-point estimate analysis.

Dynamic Programming[11]:
Dynamic programming is a method for solving complex problems by breaking them down into simpler sub-problems. It is applicable to problems exhibiting the properties of overlapping sub-problems and optimal substructure. When applicable, the method takes far less time than simple methods that don't take advantage of the sub-problem overlap. The dynamic programming approach seeks to solve each sub-problem only once, thus reducing the number of computations: once the solution to a given sub-problem has been computed, it is stored and when the next time the same solution is needed, it is simply looked up. This approach is especially useful when the number of repeating sub-problems grows exponentially as a function of the size of the input.
Dynamic programming algorithms are used for optimization. A dynamic programming algorithm will examine all possible ways to solve the problem and will pick the best solution. Therefore, dynamic programming can be thought of as an intelligent, brute-force method that enables us to go through all possible solutions to pick the best one. If the scope of the problem is such that going through all possible solutions is possible and fast enough, dynamic programming guarantees finding the optimal solution. (Wikipedia)

Bayesian Inference[10]:
In statistics, Bayesian inference is a method of inference in which Bayes' rule is used to update the probability estimate for a hypothesis as additional evidence is acquired. Bayesian updating is an important technique throughout statistics, and especially in mathematical statistics. For some cases, exhibiting a Bayesian derivation for a statistical method automatically ensures that the method works as well as any competing method. Bayesian updating is especially important in the dynamic analysis of a sequence of data.
Baye’s Rule:

$$P(H|E) = \frac{P(E|H) \times P(H)}{P(E)}$$

Bayesian inference derives the posterior probability as a consequence of two antecedents, a prior probability and a "likelihood function" derived from a probability model for the data to be observed. Bayesian inference computes the posterior probability according to Bayes' rule. The posterior predictive distribution is the distribution of a new data point, marginalized over the posterior. :

$$p(\tilde{x}|X,\alpha) = \int_{\theta} p(\tilde{x}|\theta)p(\theta\theta|X,\alpha)d\theta$$

Artificial Neutral Networks[7]:
Neural Networks are a different paradigm for computing.
Neural networks are based on the parallel architecture of animal brains. Artificial neural networks are computational models inspired by animal central nervous systems that are capable of machine learning and pattern recognition. Neural networks are a form of multiprocessor computer system, with simple processing elements, a high degree of interconnection simple scalar messages and adaptive interaction between elements .A biological neuron may have as many as 10,000 different inputs, and may send its output to many other neurons. Neurons are wired up in a 3-dimensional pattern. Neural networks, with their remarkable ability to develop meaning from complicated or inexact data, can be used to

extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques.

References:

[1] Kaun Banega Crorepati. (2013, October 16). In *Wikipedia, The Free Encyclopedia*. Retrieved 22:00, October 17, 2013, from http://en.wikipedia.org/w/index.php?title=Kaun_Banega_Crorepati&oldid=577435792
[2] J-archive - http://www.j-archive.com/help.php
[3] IBM power 750 severs, http://public.dhe.ibm.com/common/ssi/ecm/en/pod03034usen/POD03034USEN.PDF
[4] Natural language processing:

1. Natural Language Processing. (2013, October 17). In *Wikipedia, The Free Encyclopedia*. Retrieved 21:34, October 17, 2013, fromhttp://en.wikipedia.org/w/index.php?title=Natural_language_processing&oldid=577561020
2. Daniel Jurafsky and James H. Martin (2008). *Speech and Language Processing*, 2nd edition. Pearson Prentice Hall. ISBN 978-0-13-187321-6.

[5] J! Archive. [Online]. Available: http://j-archive.com/
[6] James Clune, Heuristic Evaluation Functions for General Game Playing, Department of Computer Science, The University of California, Los Angeles. Link: http://logic.stanford.edu/classes/cs227/2013/readings/cluneplayer.pdf
[7] Artificial neural network:

1. Artificial neural network. (2013, October 15). In *Wikipedia, The Free Encyclopedia*. Retrieved 22:36, October 17, 2013, romhttp://en.wikipedia.org/w/index.php?title=Artificial_neural_network&oldid=577352836
2. NEURAL NETWORKS by Christos Stergiou and Dimitrios Siganos. http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html#What is a Neural Network

[8] Monte Carlo simulations:

1. Christian P.Robert, George Casella, Introducing Monte Carlo Methods With R. ISBN : 978-1-4419-1575-7
2. Monte Carlo method. (2013, October 2). In *Wikipedia, The Free Encyclopedia*. Retrieved 22:47, October 17, 2013, from http://en.wikipedia.org/w/index.php?title=Monte_Carlo_method&oldid=575476710

[9] Best response. (2013, May 26). In *Wikipedia, The Free Encyclopedia*. Retrieved 22:35, October 17, 2013, from http://en.wikipedia.org/w/index.php?title=Best_response&oldid=556934186
[10] Bayesian Inference:

1. WILLIAM M.BOLSTAD, INTRODUCTION TO BAYESIAN STATISTICS. ISBN: 0-471-27020-2
2. Bayesian inference. (2013, September 15). In *Wikipedia, The Free Encyclopedia*. Retrieved 23:03, October 17, 2013, fromhttp://en.wikipedia.org/w/index.php?title=Bayesian_inference&oldid=573039530

[11] Dynamic Programming:

1. S. Dasgupta, C.H. Papadimitriou, and U.V. Vazirani, '**Algorithms**', p 173, available at http://www.cs.berkeley.edu/~vazirani/algorithms.html
2. Dynamic programming. (2013, October 17). In *Wikipedia, The Free Encyclopedia*. Retrieved 21:34, October 17, 2013, fromhttp://en.wikipedia.org/w/index.php?title=Dynamic_programming&oldid=577509672

Further Readings:

1. G. Tesauro, D. C. Gondek, J. Lenchner, J. Fan and J. M. Prager (2013) "Analysis of Watson's Strategies for Playing Jeopardy!", JAIR, Volume 47, pages 205-251. Link: http://www.jair.org/media/3834/live-3834-7061-jair.pdf

2. G. Tesauro, D. C. Gondek, J. Lenchner, J. Fan and J. M. Prager. 2012. Simulation, Learning and optimization techniques in Watson's game strategies. IBM journal of Research and Development