I’ve defined efficiency as the damage a spell deals or heals at the instant the spell hits divided by the cooldown of the spell. This definition favors spells that deal all of their damage in the first instant, such as Irelia’s Bladesurge. Spells, such as Swain’s Decrepify, will have a lower efficiency value because they do not deal all the damage in the first instance; it requires multiple ticks over time to deal the full amount of damage. Therefore, spells with immediate high base damage and low cooldowns will be rated higher.

According to my algorithm, Noxious Traps was the most efficient spell for various levels of attack damage, spell damage, cooldown, health, and armor. However, when attack damage got high (around 300 attack damage) Aatrox’s Blood Thirst / Blood Price became more efficient. These calculations are only accurate on paper (according to my algorithm) but not in practice. In practice, the efficiency of Blood Thirst / Blood Price should be calculated on Aatrox’s attack speed – not the cooldown of the spell, which only serves to toggle between the two states. Teemo’s Noxious traps also aren’t practically the most efficient spell, since they are on a charge system. The 0.5 second cooldown of Noxious Traps only works while Teemo has charges, but he only gets charges every 34/28/22 seconds.

Some champion spells weren’t accurately analyzed. When Lux’s spell data was retrieved from the API, the “effect” field of her spells were level tips – not the list of doubles that the API said it should be. As a result, her base damage was not added to the damage her spells dealt; instead only her scaling damage from spell damage was accounted for. Her spells’ base damage could not be found for her spells. Another limitation of the algorithm used is that it only evaluated spells that either dealt damage or spells that healed. Spells that shielded champions, granted movement speed, granted bonus health, etc were not included in the calculation of efficiency. An additional limitation is that spells are only analyzed on either their damage dealt *or* their health healed but if a spell does both, damage was prioritized and used for calculating efficiency.

The analysis of the stats is also slow. This should’ve been sped up by utilizing either parallel computing or storing more data local during the algorithm’s execution.